Recipes Research Project

Name(s): Jianqi Wu and Yuxuan Zhang

Website Link: [https://jiangqiw.github.io/Recipes-Research-Project/ (https://jiangqiw.github.io/Recipes-Research-Project/)](https://jiangqiw.github.io/Recipes-Research-Project/)

In [1]:
```
import plotly.io as pio
pio.renderers.default='notebook'
```

Code

Introduction

The world of cooking and food has grown exponentially over the years, with a vast array of recipes and cuisines available at our fingertips. In this project, we aim to dive into an extensive dataset consisting of recipes and their respective ratings. Our goal is to uncover trends and patterns that can provide valuable insights into the factors contributing to the popularity and success of a recipe.

The dataset is sourced from food.com and contains recipes and reviews posted since 2008. The data is divided into two parts: recipes and ratings. The recipes dataset includes information such as recipe name, ID, preparation time, contributor ID, submission date, tags, nutrition information, number of steps, steps text, and description. The ratings dataset, on the other hand, contains user ID, recipe ID, date of interaction, rating, and review text.

In the project, we will be first cleaning the data set and conduct exploratory data analysis, to obtain some basic information of the data set and relation between columns. Then, we will assess the missingness contained in the data set by NMAR analysis and analyzing the missingness dependency. Last, we would focus on the research question that, are complex recipes and simple recipes rated in the same scale. We would define recipe with fewer than 10 steps as simple recipes, and with more than 10 steps as complex recipes. We would analyze the rating scale related to the complexity of the recipe.

Import required package and import the data set from csv file

In [2]:
```
import pandas as pd
import numpy as np
import os
from scipy.stats import ks_2samp
import plotly.express as px
import plotly.figure_factory as ff
pd.options.plotting.backend = 'plotly'
```

In [3]:
```
recipes = pd.read_csv(os.path.join('food_data', 'RAW_recipes.csv'))
recipes.head()
```

Out[3]:

	name	id	minutes	contributor_id	submitted	tags	nutrition	n_steps	steps
0	1 brownies in the world best ever	333281	40	985201	2008-10-27	['60-minutes-or-less', 'time-to-make', 'course...	[138.4, 10.0, 50.0, 3.0, 3.0, 19.0, 6.0]	10	['heat t oven 350f a arran the ra
1	1 in canada chocolate chip cookies	453467	45	1848091	2011-04-11	['60-minutes-or-less', 'time-to-make', 'cuisin...	[595.1, 46.0, 211.0, 22.0, 13.0, 51.0, 26.0]	12	['pı he oven t 3 degre f', 'ir mix
2	412 broccoli casserole	306168	40	50969	2008-05-30	['60-minutes-or-less', 'time-to-make', 'course...	[194.8, 20.0, 6.0, 32.0, 22.0, 36.0, 3.0]	6	['prehe oven 3 degree 'spray 2 qua
3	millionaire pound cake	286009	120	461724	2008-02-12	['time-to-make', 'course', 'cuisine', 'prepara...	[878.3, 63.0, 326.0, 13.0, 20.0, 123.0, 39.0]	7	['frehe the ov to 3 degree 'grease
4	2000 meatloaf	475785	90	2202916	2012-03-06	['time-to-make', 'course', 'main-ingredient', ...	[267.0, 30.0, 12.0, 12.0, 29.0, 48.0, 2.0]	17	['pan baco and s aside a pap tov

In [4]:
```
interaction = pd.read_csv(os.path.join('food_data', 'RAW_interactions.csv'))
interaction.head()
```

Out[4]:

	user_id	recipe_id	date	rating	review
0	1293707	40893	2011-12-21	5	So simple, so delicious! Great for chilly fall...
1	126440	85009	2010-02-27	5	I made the Mexican topping and took it to bunk...
2	57222	85009	2011-10-01	5	Made the cheddar bacon topping, adding a sprin...
3	124416	120345	2011-08-06	0	Just an observation, so I will not rate. I fo...
4	2000192946	120345	2015-05-10	2	This recipe was OVERLY too sweet. I would sta...

Data cleaning

First I check the data type for each column and think about the necessary data cleaning steps.

In [5]:
```
# checking data type
recipes.dtypes
```

Out[5]:
```
name               object
id                  int64
minutes             int64
contributor_id      int64
submitted          object
tags               object
nutrition          object
n_steps             int64
steps              object
description        object
ingredients        object
n_ingredients       int64
dtype: object
```

The first step we are going to do to the dataframe is the tags, steps and ingredients columns. The three column all look like lists of string, but by checking the specific entry in the dataframe, we find that they are actually not lists. This could due to when web scraping, data collecter does not convert the text into list. As a result, we take action to convert the these three columns into list of string.

In [6]:
```
# changing columns into list
recipes['tags'] = recipes['tags'].str.strip('[').str.strip(']').str.split(',')
recipes['steps'] = recipes['steps'].str.strip('[').str.strip(']').str.split(',')
recipes['ingredients'] = recipes['ingredients'].str.strip('[').str.strip(']').str.spl
```

Then, since there are two dataframe but with common column, which are `id` and `recipe_id`. As a result, we merge the two dataframe together to show the recipes and corresponding rating and review.

In [7]:
```
# Merging two dataframe
merged = recipes.merge(interaction, left_on='id', right_on='recipe_id', how = 'left')
merged.head()
```

Out[7]:

	name	id	minutes	contributor_id	submitted	tags	nutrition	n_steps	steps
0	1 brownies in the world best ever	333281	40	985201	2008-10-27	['60-minutes-or-less', 'time-to-make', 'cour...	[138.4, 10.0, 50.0, 3.0, 3.0, 19.0, 6.0]	10	['heat the oven to 350f and arrange the rack i...
1	1 in canada chocolate chip cookies	453467	45	1848091	2011-04-11	['60-minutes-or-less', 'time-to-make', 'cuis...	[595.1, 46.0, 211.0, 22.0, 13.0, 51.0, 26.0]	12	['pre-heat oven the 350 degrees f', 'in a mix...
2	412 broccoli casserole	306168	40	50969	2008-05-30	['60-minutes-or-less', 'time-to-make', 'cour...	[194.8, 20.0, 6.0, 32.0, 22.0, 36.0, 3.0]	6	['preheat oven to 350 degrees', 'spray a 2 qu...
3	412 broccoli casserole	306168	40	50969	2008-05-30	['60-minutes-or-less', 'time-to-make', 'cour...	[194.8, 20.0, 6.0, 32.0, 22.0, 36.0, 3.0]	6	['preheat oven to 350 degrees', 'spray a 2 qu...
4	412 broccoli casserole	306168	40	50969	2008-05-30	['60-minutes-or-less', 'time-to-make', 'cour...	[194.8, 20.0, 6.0, 32.0, 22.0, 36.0, 3.0]	6	['preheat oven to 350 degrees', 'spray a 2 qu...

We find in the interaction dataframe, one important data is the rating for the recipes. As a result, we add new column name `ave_rating`, which include the average rating for the column. Also, we beleve that the 0 in the rating might be empty rating that people do not fill in. As a result, we replace 0 with nan value

In [8]:
```
# Add average rating column and replace 0 with nan
ser = merged.groupby('id').agg({'rating': 'mean'}).replace(0, np.nan)['rating']
recipes = recipes.set_index('id')
recipes['ave_rating'] = ser
recipes = recipes.reset_index()
```

We also find that the `nutrition` column in the dataframe look like a list containing float but actually not. We find that in the list, the float represent: `'calories', 'total fat (PDV)', 'sugar (PDV)', 'sodium (PDV)', 'protein (PDV)', 'saturated fat (PDV)', 'carbohydrates (PDV)'` . As a result, we first convert the column into list of float and create individual column for each nutrition

In [9]:
```
# changing nutrition into column and create individual columns for each nutrition
recipes['nutrition'] = recipes['nutrition'].str.strip('[').str.strip(']').str.split(',
nutrient_names = ['calories', 'total fat (PDV)', 'sugar (PDV)', 'sodium (PDV)', 'prot
for index, nutrient in enumerate(nutrient_names):
    recipes[nutrient] = recipes['nutrition'].apply(lambda x: x[index])
    recipes[nutrient] = pd.to_numeric(recipes[nutrient], errors='coerce')
```

In [10]:
```
# Changing submitted column into datetime
recipes['submitted'] = pd.to_datetime(recipes['submitted'])
```

In [11]:
```
recipes.head()
```

Out[11]:

	id	name	minutes	contributor_id	submitted	tags	nutrition	n_steps	steps
0	333281	1 brownies in the world best ever	40	985201	2008-10-27	['60-minutes-or-less', 'time-to-make', 'cour...	[138.4, 10.0, 50.0, 3.0, 3.0, 19.0, 6.0]	10	['heat ove 350f arra the r
1	453467	1 in canada chocolate chip cookies	45	1848091	2011-04-11	['60-minutes-or-less', 'time-to-make', 'cuis...	[595.1, 46.0, 211.0, 22.0, 13.0, 51.0, 2...	12	[' h oven degr f', ' m
2	306168	412 broccoli casserole	40	50969	2008-05-30	['60-minutes-or-less', 'time-to-make', 'cour...	[194.8, 20.0, 6.0, 32.0, 22.0, 36.0, 3.0]	6	['preh ove degre 'spra 2 c
3	286009	millionaire pound cake	120	461724	2008-02-12	['time-to-make', 'course', 'cuisine', 'prep...	[878.3, 63.0, 326.0, 13.0, 20.0, 123.0, ...	7	['freh the o to degre 'gre
4	475785	2000 meatloaf	90	2202916	2012-03-06	['time-to-make', 'course', 'main-ingredient'...	[267.0, 30.0, 12.0, 12.0, 29.0, 48.0, 2.0]	17	['par bac and aside a pa

In [12]:
```
recipes.dtypes
```

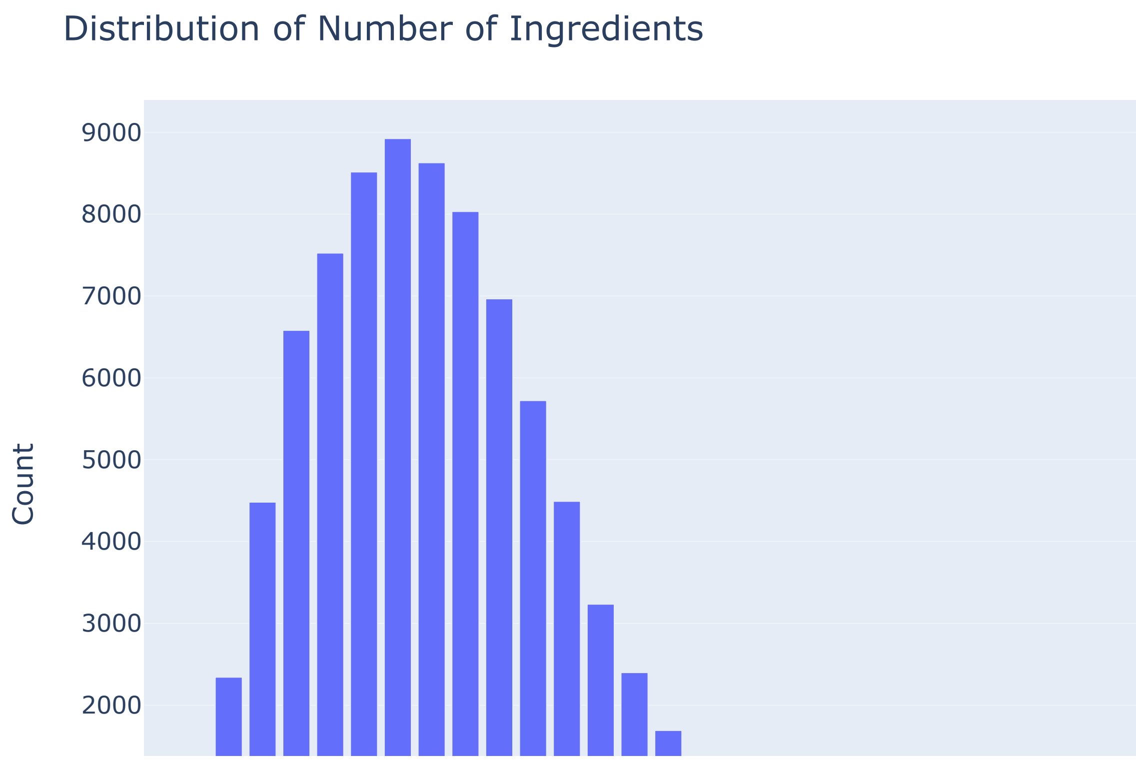
Out[12]:
```
id                             int64
name                          object
minutes                        int64
contributor_id                 int64
submitted             datetime64[ns]
tags                          object
nutrition                     object
n_steps                        int64
steps                         object
description                   object
ingredients                   object
n_ingredients                  int64
ave_rating                   float64
calories                     float64
total fat (PDV)              float64
sugar (PDV)                  float64
sodium (PDV)                 float64
protein (PDV)                float64
saturated fat (PDV)          float64
carbohydrates (PDV)          float64
dtype: object
```

In [13]:
```
df_display = recipes.drop(['description'], axis = 1)
#print(df_display.head(3).to_markdown(index=False))
```

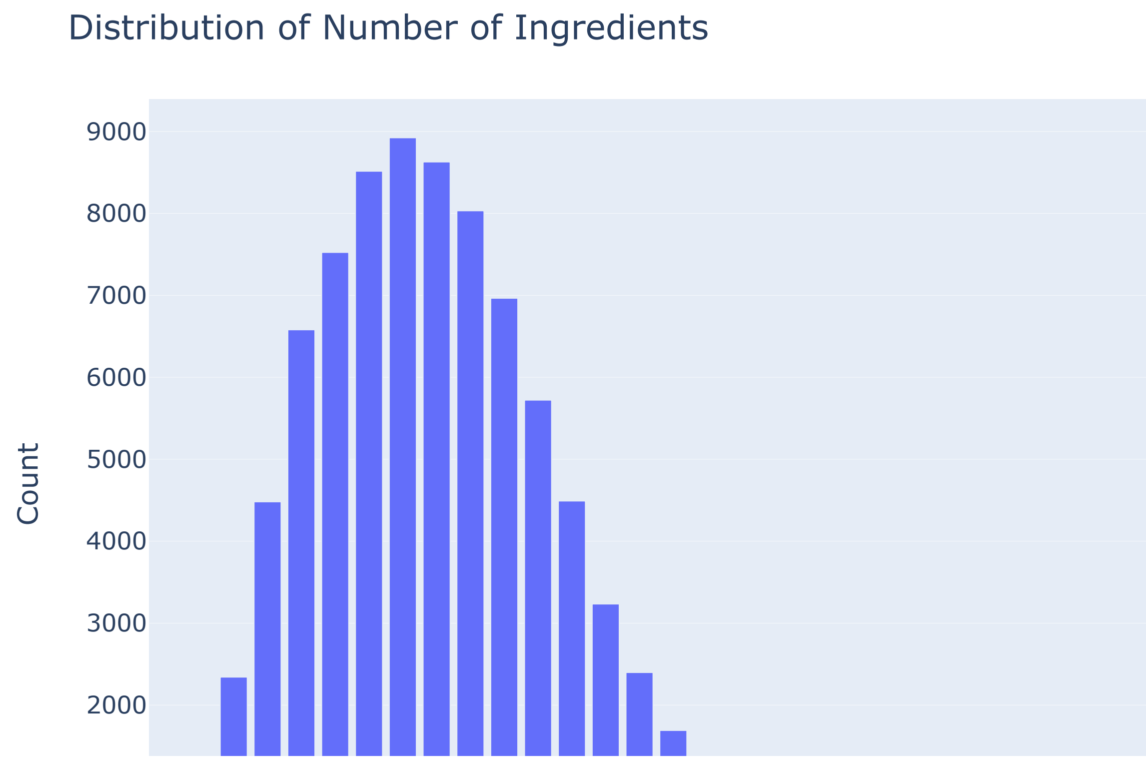
EDA

Frist we would analyze the distribution of number of ingredients

In [14]:
```
df = recipes.groupby('n_ingredients').count().reset_index()
fig1 = px.bar(df, x = 'n_ingredients', y = 'name')
fig1.update_yaxes(title='Count')
fig1.update_layout(title='Distribution of Number of Ingredients')
```



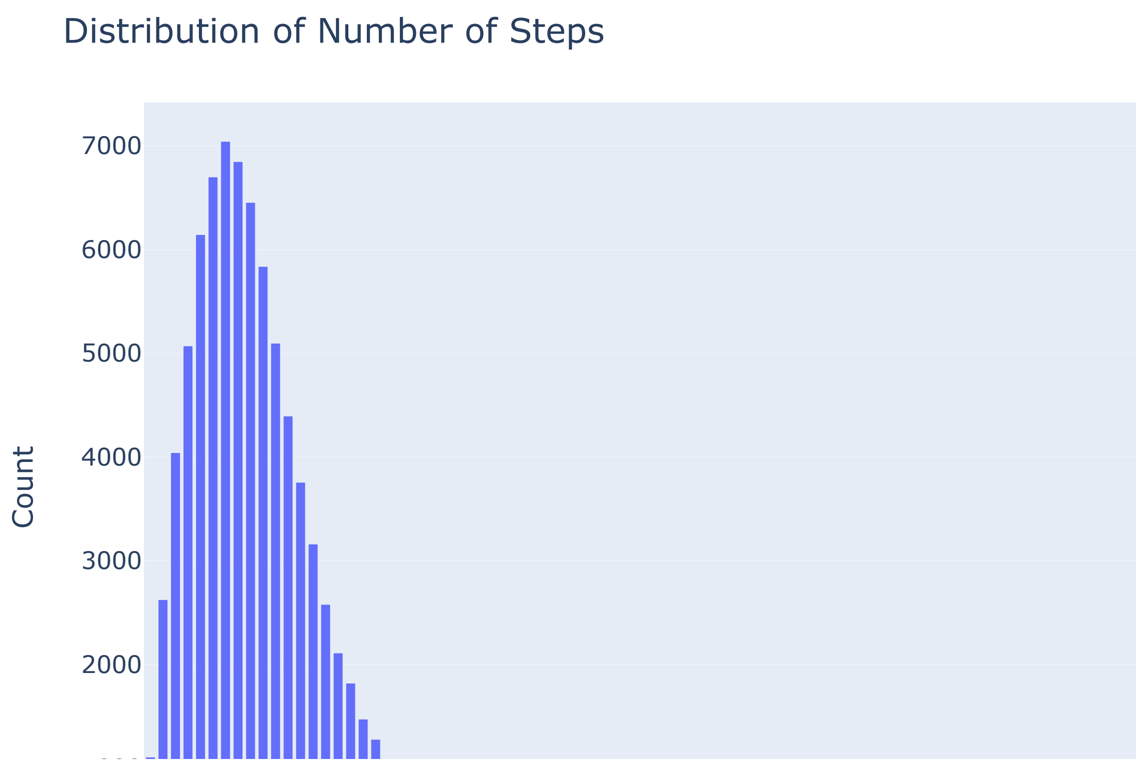
In [15]:
```
fig1.show('notebook')
```



This shows that the distribution could be approximate as a gaussian distribution but skewed right. We would say that the graph centered around 8, meaning that most recipes have 8 ingredients.

Then we analyze the distribution of number of steps

In [16]:
```
df1 = recipes.groupby('n_steps').count().reset_index()
fig2 = px.bar(df1, x = 'n_steps', y = 'name')
fig2.update_yaxes(title='Count')
fig2.update_layout(title='Distribution of Number of Steps')
fig2.show()
```



The distrubution also show similar trend in the number of steps, which is a right skewed gaussian distribution. By comparing at the two graph, the graph for the number of distribution is more centered. The center for the graph is around 7, meaning most recipes have 7 steps. Also, we could see the graph have a lot outliers that have very big step numbers. After observing the dataset and also consider together with the `minutes` column and real life situation, we decided to choose steps greater than 40 and minutes greater than 200 as outlier and not faithful data

Then, we do bivariate analysis between the number of steps and the number of ingredients

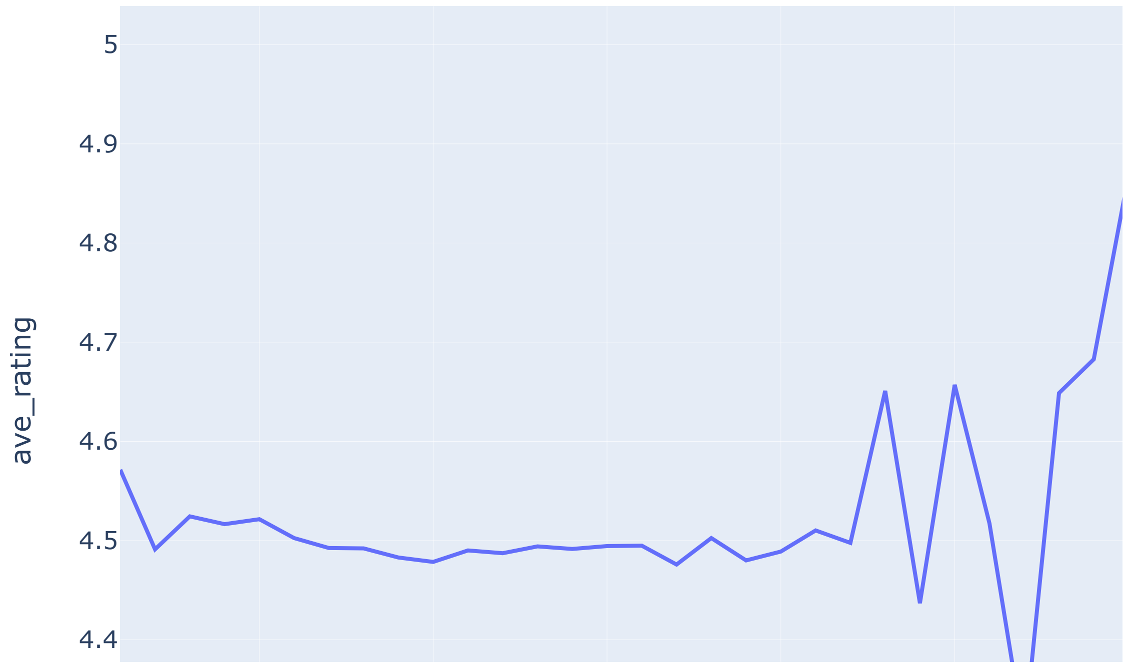
In [17]:
```
fig3 = px.scatter(recipes, x = 'n_steps', y = 'n_ingredients')

fig3.show()
```

When the individual distritbution for number of steps and number of ingredients seems very similar, the scatter plot does not know very strong correlation between the number of steps and number of ingredient. We could say that there is weak positive relationship between the number of steps and the number of ingredients.

Then, we draw line graph to present the relationship between number of ingredients and the average rating of recipe

In [18]:
```
df2 = recipes.groupby('n_ingredients').mean().reset_index()
fig4 = px.line(df2, x = 'n_ingredients', y = 'ave_rating')
fig4.show()
```



We could see that the average rating and the number of ingredients in the recipes do not have much relationship with each other. Especially with number of ingredients smaller than 15, it is almost a horizontal line, showing no relationship between the two variables. The large fluctuate with number of ingredients larger than 15 could be due to relatively small data size collected within that range.

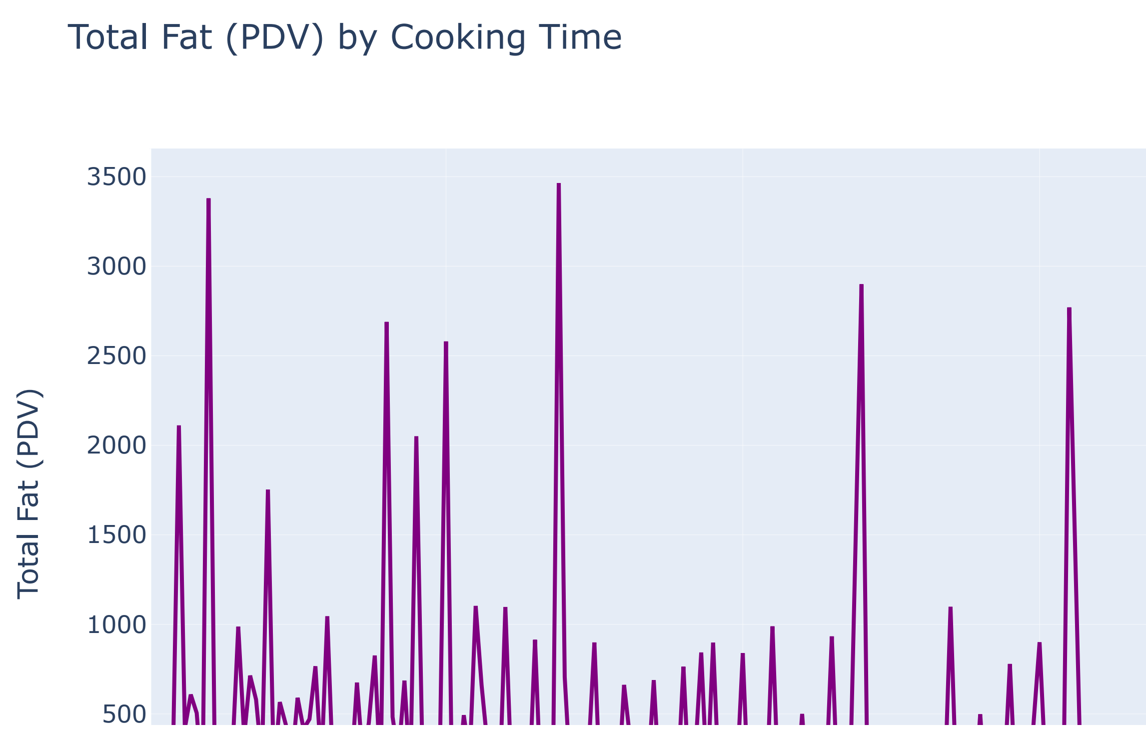
Interesting Aggregates: Analyzing the total fat with the cooking minutes

In [19]:
```
import plotly.graph_objs as go
recipes_df = recipes.copy()

recipes_df = recipes_df[recipes_df['minutes'] <= 200] # get rid of outliers
pivot_table = recipes_df.pivot_table(values='total fat (PDV)', index='minutes', aggfu
pivot_table = pivot_table.reset_index()
fig6 = go.Figure()

fig6.add_trace(go.Scatter(x=pivot_table['minutes'], y=pivot_table[('mean', 'total fat
fig6.add_trace(go.Scatter(x=pivot_table['minutes'], y=pivot_table[('median', 'total f
fig6.add_trace(go.Scatter(x=pivot_table['minutes'], y=pivot_table[('min', 'total fat
fig6.add_trace(go.Scatter(x=pivot_table['minutes'], y=pivot_table[('max', 'total fat

fig6.update_layout(title='Total Fat (PDV) by Cooking Time', xaxis_title='Cooking Time
fig6.show()
```



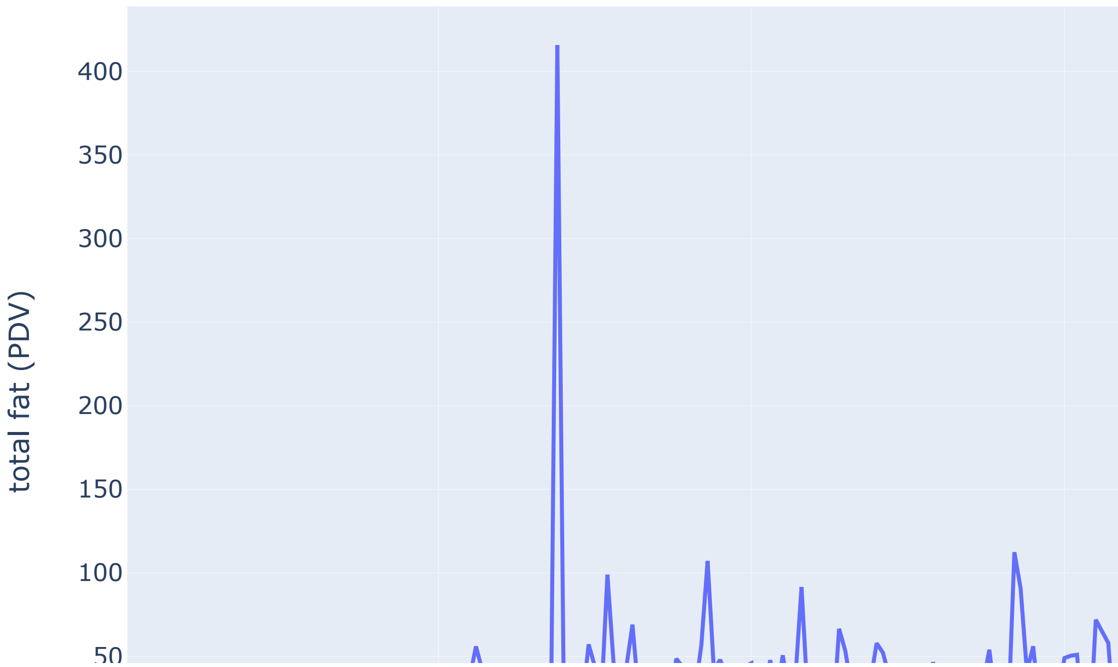
In [20]:
```
pivot_table
```

Out[20]:

	minutes	mean	median	min	max
		total fat (PDV)	total fat (PDV)	total fat (PDV)	total fat (PDV)
0	0	46.000000	46.0	46.0	46.0
1	1	7.786026	0.0	0.0	159.0
2	2	9.690529	0.0	0.0	419.0
3	3	12.579381	2.0	0.0	411.0
4	4	20.471910	7.0	0.0	258.0
...	...	...	...	...	...
182	192	36.333333	24.0	4.0	81.0
183	193	14.000000	14.0	14.0	14.0
184	195	37.130252	21.0	0.0	455.0
185	198	27.000000	27.0	27.0	27.0
186	200	41.764706	21.0	0.0	455.0

187 rows × 5 columns

In [21]:
```
df4 = recipes_df.groupby('minutes').mean().reset_index()
fig7 = px.line(df4, x = 'minutes', y = 'total fat (PDV)')
# Show the chart
fig7.show()
```



One interesting result that we find in the aggregates data is that there is a peek for total fat in the recipe around 60 minutes of cooking time. Otherwise the recipes' total fat is fluctuate around 50 PDV, which is around 1000 calories. This shows that most recipes collected are recipes for health food.

Assessment of Missingness

In [22]:
```
###### a lot of objects in columns
merged_df = recipes.merge(interaction, left_on='id', right_on='recipe_id', how = 'lef
merged_df = merged_df.drop('ave_rating', axis = 1)
### missing description rating review
merged_df['rating'] = merged_df['rating'].replace(0, np.nan)
merged_df.info()
merged_df.head()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 234429 entries, 0 to 234428
Data columns (total 24 columns):
 #   Column              Non-Null Count   Dtype
---  ------              --------------   -----
 0   id                  234429 non-null  int64
 1   name                234428 non-null  object
 2   minutes             234429 non-null  int64
 3   contributor_id      234429 non-null  int64
 4   submitted           234429 non-null  datetime64[ns]
 5   tags                234429 non-null  object
 6   nutrition           234429 non-null  object
 7   n_steps             234429 non-null  int64
 8   steps               234429 non-null  object
 9   description         234315 non-null  object
 10  ingredients         234429 non-null  object
 11  n_ingredients       234429 non-null  int64
 12  calories            234429 non-null  float64
 13  total fat (PDV)     234429 non-null  float64
 14  sugar (PDV)         234429 non-null  float64
 15  sodium (PDV)        234429 non-null  float64
 16  protein (PDV)       234429 non-null  float64
 17  saturated fat (PDV) 234429 non-null  float64
 18  carbohydrates (PDV) 234429 non-null  float64
 19  user_id             234428 non-null  float64
 20  recipe_id           234428 non-null  float64
 21  date                234428 non-null  object
 22  rating              219393 non-null  float64
 23  review              234371 non-null  object
dtypes: datetime64[ns](1), float64(10), int64(5), object(8)
memory usage: 44.7+ MB
```

Out[22]:

	id	name	minutes	contributor_id	submitted	tags	nutrition	n_steps	steps
0	333281	1 brownies in the world best ever	40	985201	2008-10-27	['60-minutes-or-less', 'time-to-make', 'cour...	[138.4, 10.0, 50.0, 3.0, 3.0, 19.0, 6.0]	10	['heat the oven to 350f and arrange the rack i...
1	453467	1 in canada chocolate chip cookies	45	1848091	2011-04-11	['60-minutes-or-less', 'time-to-make', 'cuis...	[595.1, 46.0, 211.0, 22.0, 13.0, 51.0, 2...	12	['pre-heat oven the 350 degrees f', 'in a mix...
2	306168	412 broccoli casserole	40	50969	2008-05-30	['60-minutes-or-less', 'time-to-make', 'cour...	[194.8, 20.0, 6.0, 32.0, 22.0, 36.0, 3.0]	6	['preheat oven to 350 degrees', 'spray a 2 qu...
3	306168	412 broccoli casserole	40	50969	2008-05-30	['60-minutes-or-less', 'time-to-make', 'cour...	[194.8, 20.0, 6.0, 32.0, 22.0, 36.0, 3.0]	6	['preheat oven to 350 degrees', 'spray a 2 qu...
4	306168	412 broccoli casserole	40	50969	2008-05-30	['60-minutes-or-less', 'time-to-make', 'cour...	[194.8, 20.0, 6.0, 32.0, 22.0, 36.0, 3.0]	6	['preheat oven to 350 degrees', 'spray a 2 qu...

5 rows × 24 columns

In [23]:

```
def create_kde_plotly(df, group_col, group1, group2, vals_col, title=''):
    fig = ff.create_distplot(
        hist_data=[df.loc[df[group_col] == group1, vals_col], df.loc[df[group_col] ==
        group_labels=[group1, group2],
        show_rug=False, show_hist=False,
        colors=['#ef553b', '#636efb'],
    )
    return fig.update_layout(title=title)
```

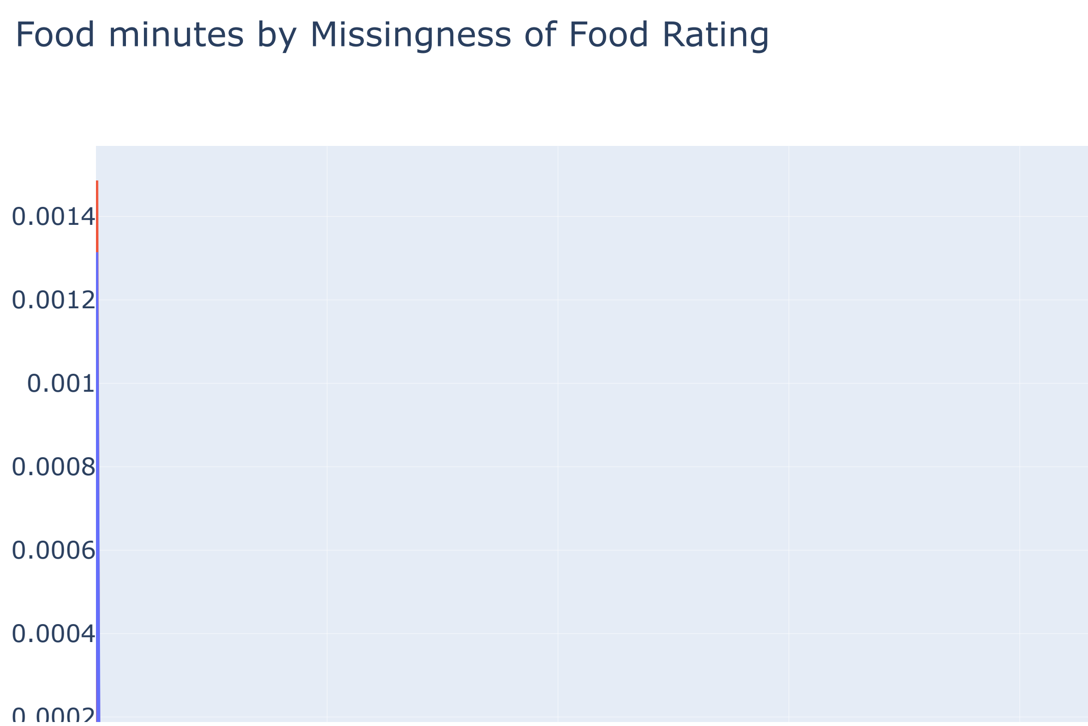
In [24]:
```
merged_df.select_dtypes(include=['int64', 'float64']).columns
```

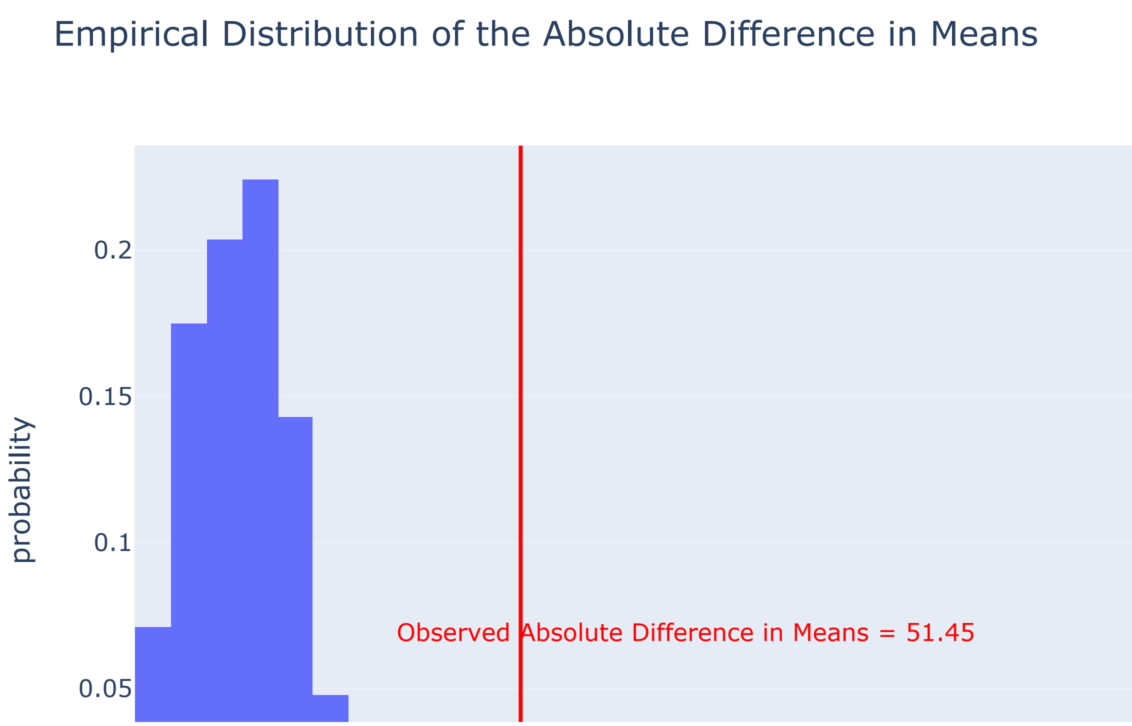
Out[24]:
```
Index(['id', 'minutes', 'contributor_id', 'n_steps', 'n_ingredients',
       'calories', 'total fat (PDV)', 'sugar (PDV)', 'sodium (PDV)',
       'protein (PDV)', 'saturated fat (PDV)', 'carbohydrates (PDV)',
       'user_id', 'recipe_id', 'rating'],
      dtype='object')
```

In [25]:
```
### MAR and MCAR Dependency Testing
## n_steps and rating dependency
def dependency_test(column_name, num, merged_df):
    merged_df = merged_df.copy()
#     iqr = merged_df[column_name].quantile(0.75) - merged_df[column_name].quantile(0.
#     threshold = 1.5*iqr + merged_df[column_name].quantile(0.25)
#     merged_df = merged_df.loc[merged_df[column_name] <= threshold, :]
    print(f'########{column_name}########')
    true_diff = abs(merged_df.loc[merged_df['rating'].isna(), column_name].mean() - m
    simulate_diff = []
    for k in range(1000):
        merged_df['shuffle_rating'] = np.random.permutation(merged_df['rating'])
        temp_diff =  abs(merged_df.loc[merged_df['shuffle_rating'].isna(), column_nam
        simulate_diff.append(temp_diff)
    print((simulate_diff >= true_diff).mean())
    fig2 = px.histogram(pd.DataFrame(simulate_diff), x=0, nbins=50, histnorm='probabi
                       title='Empirical Distribution of the Absolute Difference in Mea
    fig2.add_vline(x=true_diff, line_color='red')
    fig2.add_annotation(text=f'<span style="color:red">Observed Absolute Difference i
                       x=1.45 * true_diff, showarrow=False, y=0.07)
    ks_test = ks_2samp(merged_df.loc[merged_df['rating'].isna(), column_name], merged_
    print(ks_test.pvalue)
    merged_df['missing_rating'] = merged_df['rating'].isna()
    fig = create_kde_plotly(merged_df, 'missing_rating', True, False, column_name,
                            f"Food {column_name} by Missingness of Food Rating")
    fig.show()
    fig2.show()
    fig.write_html(f'fig{num}.html', include_plotlyjs='cdn')
    fig2.write_html(f'fig{num + 1}.html', include_plotlyjs='cdn')
```

In [34]:
```
dependency_test('minutes', 11, merged_df)
```

```
########minutes########
0.124
1.4191797241819564e-107
```

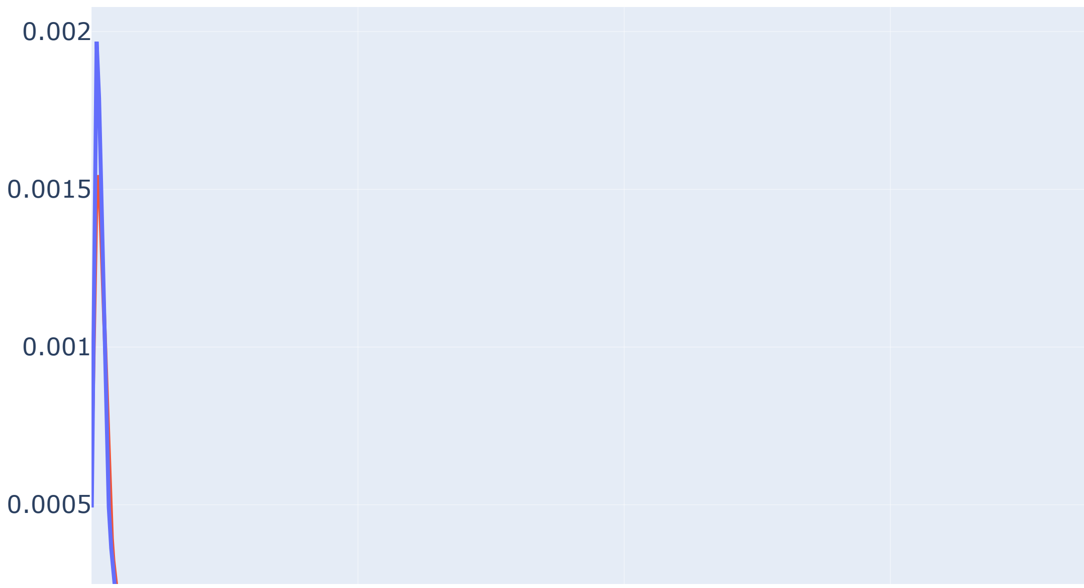


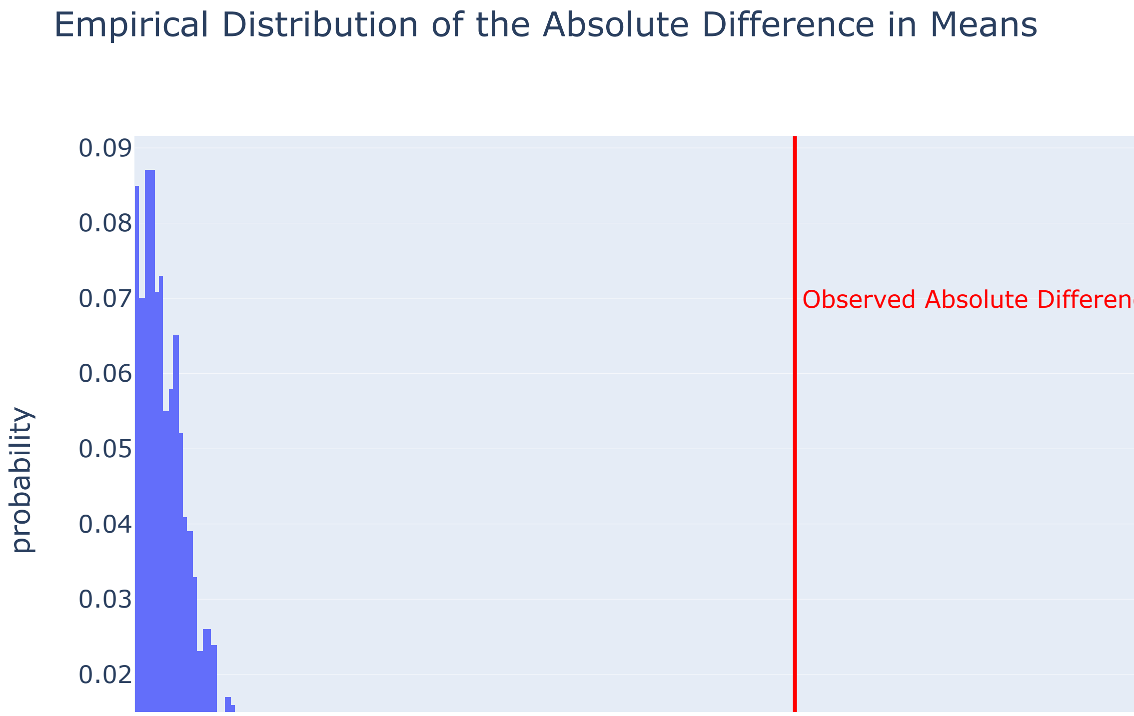

Empirical Distribution of the Absolute Difference in Means
probability
0.2
0.15
0.1
0.05
Observed Absolute Difference in Means = 51.45

In [27]:
```
dependency_test('calories', 13, merged_df)
```

```
########calories########
0.0
7.960756339494917e-35
```

Food calories by Missingness of Food Rating




Empirical Distribution of the Absolute Difference in Means
probability
0.09
0.08
0.07
0.06
0.05
0.04
0.03
0.02
Observed Absolute Differenc

Hypothesis Testing

In [28]:
```
recipes
```

Out[28]:

	id	name	minutes	contributor_id	submitted	tags	nutrition	n_steps
0	333281	1 brownies in the world best ever	40	985201	2008-10-27	['60-minutes-or-less', 'time-to-make', 'cour...	[138.4, 10.0, 50.0, 3.0, 3.0, 19.0, 6.0]	10
1	453467	1 in canada chocolate chip cookies	45	1848091	2011-04-11	['60-minutes-or-less', 'time-to-make', 'cuis...	[595.1, 46.0, 211.0, 22.0, 13.0, 51.0, 2...	12
2	306168	412 broccoli casserole	40	50969	2008-05-30	['60-minutes-or-less', 'time-to-make', 'cour...	[194.8, 20.0, 6.0, 32.0, 22.0, 36.0, 3.0]	6
3	286009	millionaire pound cake	120	461724	2008-02-12	['time-to-make', 'course', 'cuisine', 'prep...	[878.3, 63.0, 326.0, 13.0, 20.0, 123.0, ...	7
4	475785	2000 meatloaf	90	2202916	2012-03-06	['time-to-make', 'course', 'main-ingredient'...	[267.0, 30.0, 12.0, 12.0, 29.0, 48.0, 2.0]	17
...	...	...	...	...	...	...	...	...
83777	486161	zydeco soup	60	227978	2012-08-29	['ham', '60-minutes-or-less', 'time-to-make'...	[415.2, 26.0, 34.0, 26.0, 44.0, 21.0, 15.0]	7
83778	493372	zydeco spice mix	5	1500678	2013-01-09	['15-minutes-or-less', 'time-to-make', 'cour...	[14.8, 0.0, 2.0, 58.0, 1.0, 0.0, 1.0]	1
83779	308080	zydeco ya ya deviled eggs	40	37779	2008-06-07	['60-minutes-or-less', 'time-to-make', 'cour...	[59.2, 6.0, 2.0, 3.0, 6.0, 5.0, 0.0]	7

	id	name	minutes	contributor_id	submitted	tags	nutrition	n_steps
83780	298512	cookies by design cookies on a stick	29	506822	2008-04-15	['30-minutes-or-less', 'time-to-make', 'cour...	[188.0, 11.0, 57.0, 11.0, 7.0, 21.0, 9.0]	9
83781	298509	cookies by design sugar shortbread cookies	20	506822	2008-04-15	['30-minutes-or-less', 'time-to-make', 'cour...	[174.9, 14.0, 33.0, 4.0, 4.0, 11.0, 6.0]	5

83782 rows × 20 columns

The question we are going to research on is that: are regular recipes and complex recipes are rated in the same scale?

In this part, we will define a complex recipes as recipes have greater than 10 steps. We will conduct a permutation test.

Null Hypothesis H0: People are rating all the recipes in the same scale.

Alternative Hypothesis H1: People are giving complex recipe lower rating

The reason for choosing one-sided test is that we might assume people could feel frustrated when cooking complex recipes, and also recipes with more steps are harder to cook

In [29]:

```
# keep only useful column, including n_steps and average rating
df_testing = recipes[['id', 'n_steps', 'ave_rating']]
df_testing = df_testing.dropna()
df_testing['complex'] = df_testing['n_steps'] > 10
obs_df = df_testing.groupby('complex').mean()
obs_df
```

Out[29]:

	id	n_steps	ave_rating
complex			
False	378151.284947	6.357180	4.501838
True	385033.935788	16.141415	4.484409

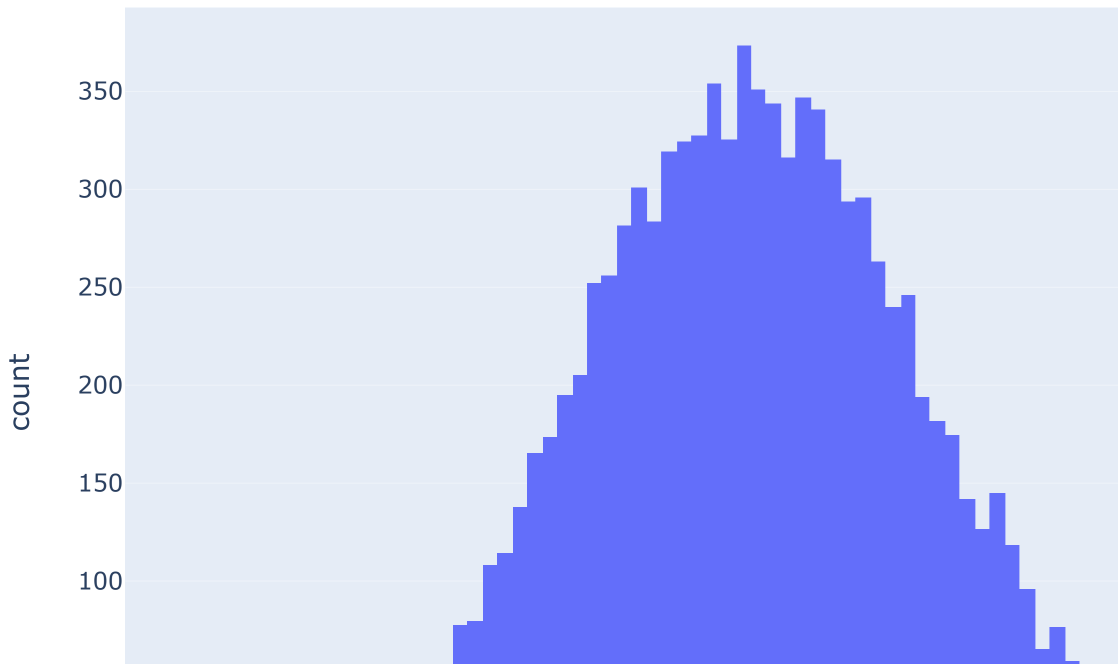
Since ave_rating is numerical data, so it is proper to use the difference in mean,

In [30]:
```
obs = obs_df['ave_rating'].iloc[0] - obs_df['ave_rating'].iloc[1]
obs
```

Out[30]: 0.017428379224658563

In [31]:
```
lst = []
for i in range(10000):
    df_testing = df_testing.assign(permuted = np.random.permutation(df_testing['compl
    temp = df_testing.groupby('permuted').mean()
    diff = temp['ave_rating'].iloc[0] - temp['ave_rating'].iloc[1]
    lst.append(diff)
arr = np.array(lst)
```

In [32]:
```
fig10 = px.histogram(pd.DataFrame(lst), x=0, nbins=100)
fig10.add_shape(
    type='line',
    x0=obs,
    x1=obs,
    y0=0,
    y1=1,
    yref='paper',  # Use relative coordinates for y (0 to 1)
    line=dict(color='red')
)
fig10.show()
```



In [33]:
```
p_value = (arr > obs).mean()
p_value
```

Out[33]: 0.0009

Since the p-value is larger than the significant level, which is 0.05, we fail to reject the null hypothesis. This result could be reasonable since first, the complexity of a recipes does not decide whether the food is delicious or not. The taste of the food, which is one important part of the rating, is not likely to be determined by the complexity of recipe. On the other hand, another possible explanation could be different people might have various opinions towards the complexity of the recipe. Some might prefer simple recipes since they are convenient and time-saving. Others could love the process of cooking and enjoy working a complex recipes.

In [ ]:

```

```