Data Curation Project

1.Introduce

1.1 Source and Inspiration

This dataset includes information on food choices, nutrition, preferences, childhood favorites, and other information from college students. There were 126 responses from students. Data is raw and uncleaned.

How important is nutrition information for today's college kids? Is their taste in food defined by their food preferences when they were children? Are kids of parents who cook more likely to make better food choices than others? Are these kids likely to have a different taste compared to others? There a number of open ended questions included in this dataset such as: What is your favorite comfort food? What is your favorite cuisine? that could work well for natural language processing.

1.2 Properties of the Data Set

We downloaded the csv file for analysis, linked below:
https://www.kaggle.com/datasets/borapajo/food-choices/download?datasetVersionNumber=5
The data set size is approximately 5.49M and the total number of records is 125.

Here are the columns of the dataset:

Column name	Desciption
GPA	actual GPA
Gender	1 – Female and 2 – Male
Breakfast	1 – cereal option and 2 – donut option
calories_chicken	guessing calories in chicken piadina 1 - 265 2 - 430 3 - 610 4 – 720 The variable shows the actual number of calories participants selected

calories_day	Importance of consuming calories per day 1 - i dont know how many calories i should consume 2 - it is not at all important 3 - it is moderately important 4 - it is very important
calories_scone	Guessing calories in a scone from starbucks 1 - 107 cal 2 - 315 cal 3 - 420 cal 4 - 980 cal (the variable shows the actual number of calories participants selected)
coffee	which of the two pictures you associate with the word coffee? 1 – creamy frapuccino 2 – espresso shown
comfort_food	List 3-5 comfort foods that come to mind. Open ended (perfect for NLP)
comfort_food_reasons	What are some of the reasons that make you eat comfort food? (i.e., anger, sadness, happiness, boredom, etc) - list up to three Open ended (perfect for NLP)
comfort_food_reasons_coded	1 – stress 2 – boredom 3 – depression/sadness 4 – hunger 5 – laziness 6 – cold weather 7 – happiness 8- watching tv 9 – none
cook	how often do you cook? 1 - Every day 2 - A couple of times a week

	3 - Whenever I can, but that is not very often 4 - I only help a little during holidays 5 - Never, I really do not know my way around a kitchen
cuisine	what type of cuisine did you eat growing up? 1 – American 2 – Mexican.Spanish 3 – Korean/Asian 4 – Indian 5 – American inspired international dishes 6 – other
diet_current	describe your current diet open ended – ideal for NLP
diet_current_coded	1 – healthy/balanced/moderated/ 2 – unhealthy/cheap/too much/random/ 3 – the same thing over and over 4 – unclear
drink	1 – orange juice 2 – soda
eating_changes	Describe your eating changes since the moment you got into college? Open ended
eating_changes_coded1	1 – worse 2 – better 3 – the same 4 – unclear
eating_changes_coded2	1 – eat faster 2 – bigger quantity 3 – worse quality 4 – same food 5 – healthier 6 – unclear 7 – drink coffee 8 – less food

	9 – more sweets 10 – timing 11 – more carbs or snacking 12 – drink more water 13 – more variety
eating_out	frequency of eating out in a typical week 1 - Never 2 - 1-2 times 3 - 2-3 times 4 - 3-5 times 5 - every day
employment	do you work? 1 - yes full time 2 - yes part time 3 – no 4 - other
ethnic_food	How likely to eat ethnic food 1 - very unlikely 2 - unlikely 3 - neutral 4 - likely 5 - very likely
exercise	how often do you exercise in a regular week? 1 - Everyday 2 - Twice or three times per week 3 - Once a week 4 - Sometimes 5 – Never
father_education	1 - less than high school 2 - high school degree 3 - some college degree 4 - college degree

	5 - graduate degree
father_profession	what is your father's profession? Open ended
fav_cuisine	What is your favorite cuisine? Open ended
fav_cuisine_coded	0-none 1 – Italian/French/greek 2 – Spanish/mexican 3 – Arabic/Turkish 4 – asian/chineses/thai/nepal 5 – American 6 – African 7 – Jamaican 8 – indian
fav_food	was your favorite food cooked at home or store bought? 1 - cooked at home 2 - store bought 3 - both bought at store and cooked at home
food_childhood	what was your favorite childhood food? Open ended
fav_food	which of these pictures you associate with word fries? 1 – Mcdonald’ s fries 2 – home fries
fruit_day	How likely to eat fruit in a regular day 1 - very unlikely 2 - unlikely 3 - neutral 4 - likely 5 - very likely
grade_level	1 - freshman 2 -Sophomore

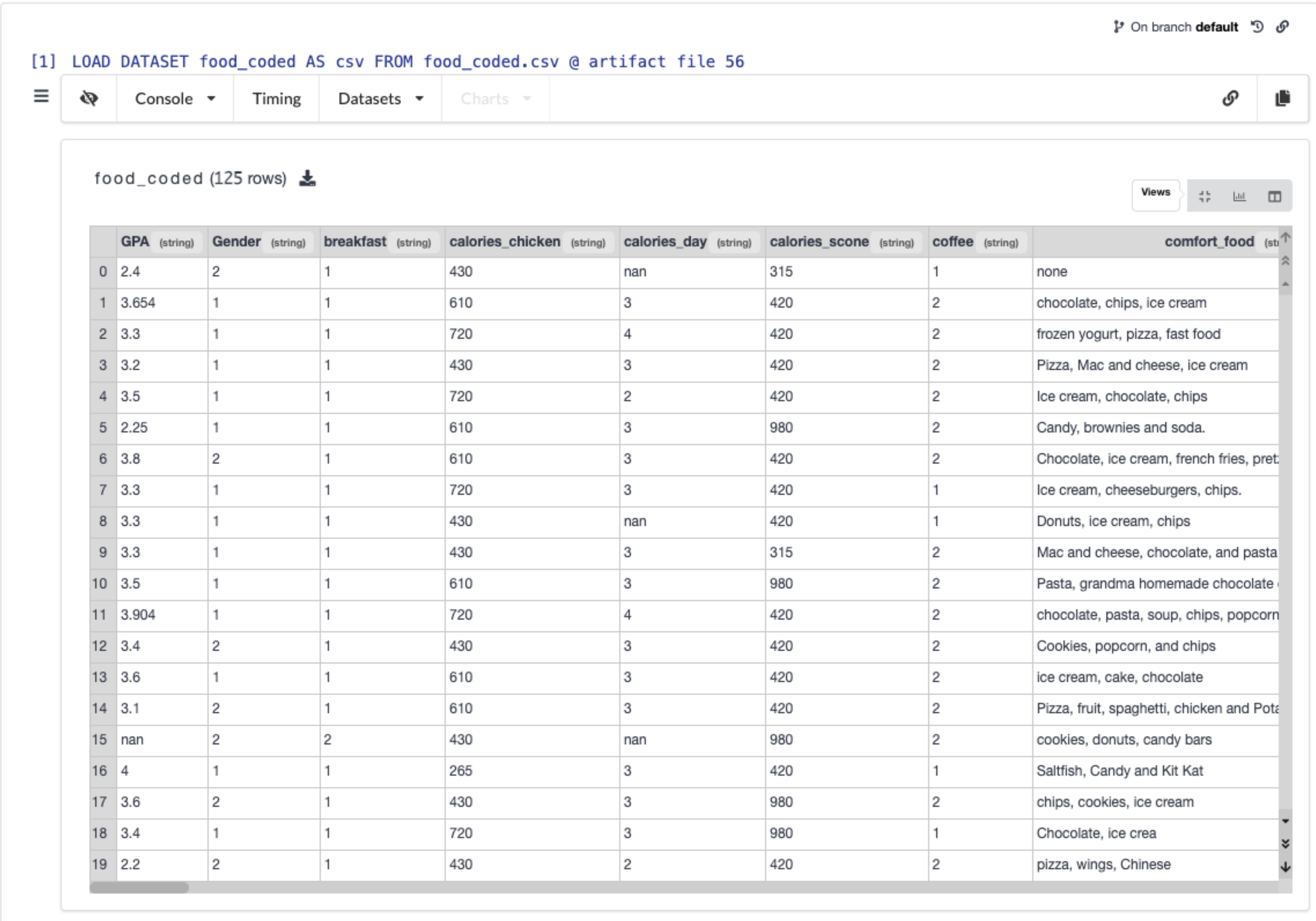
	3 - Junior 4 - Senior
greek_food	How likely to eat greek food when available? 1 - very unlikely 2 - unlikely 3 - neutral 4 - likely 5 - very likely
healthy_feel	how likely are you to agree with the following statement: “I feel very healthy!” ? 1 to 10 where 1 is strongly agree and 10 is strongly disagree - scale
healthy_meal	what is a healthy meal? Describe in 2-3 sentences. Open ended
ideal_diet	describe your ideal diet in 2-3 sentences Open ended
Ideal_diet_coded	1 – portion control 2 – adding veggies/eating healthier food/adding fruit 3 – balance 4 – less sugar 5 – home cooked/organic 6 – current diet 7 – more protein 8 – unclear
income	1 - less than $15,000 2 - $15,001 to $30,000 3 - $30,001 to $50,000 4 - $50,001 to $70,000 5 - $70,001 to $100,000 6 - higher than $100,000
indian_food	how likely are you to eat indian food when available 1 - very unlikely

	2 - unlikely 3 - neutral 4 - likely 5 - very likely
Italian_food	how likely are you to eat Italian food when available? 1 - very unlikely 2 - unlikely 3 - neutral 4 - likely 5 - very likely
life_rewarding	how likely are you to agree with the following statement: “I feel life is very rewarding!” ? 1 to 10 where 1 is strongly agree and 10 is strongly disagree - scale
marital_status	1 -Single 2 - In a relationship 3 - Cohabiting 4 - Married 5 - Divorced 6 - Widowed
meals_dinner_friend	What would you serve to a friend for dinner? Open ended 43) mothers_education 1 - less than high school 2 - high school degree 3 - some college degree 4 - college degree 5 - graduate degree
mothers_profession	what is your mother’s profession?
nutritional_check	checking nutritional values frequency 1 - never

	2 - on certain products only 3 - very rarely 4 - on most products 5 - on everything
on_off_campus	living situation 1 - On campus 2 - Rent out of campus 3 - Live with my parents and commute 4 - Own my own house
parents_cook	Approximately how many days a week did your parents cook? 1 - Almost everyday 2 - 2-3 times a week 3 - 1-2 times a week 4 - on holidays only 5 - never
pay_meal_out	How much would you pay for meal out? 1 - up to $5.00 2 - $5.01 to $10.00 3 - $10.01 to $20.00 4 - $20.01 to $30.00 5 - $30.01 to $40.00 6 - more than $40.01
Persian_food	1 - very unlikely 2 - unlikely 3 - neutral 4 - likely 5 - very likely
self_perception_weight	self perception of weight 6 - i dont think myself in these terms 5 - overweight 4 - slightly overweight 3 - just right

	2 - very fit 1 - slim
soup	Which of the two pictures you associate with the word soup? 1 – veggie soup 2 – creamy soup
sports	do you do any sporting activity? 1 - Yes 2 - No 99 – no answer
thai_food	How likely to eat thai food when available? 1 - very unlikely 2 - unlikely 3 - neutral 4 - likely 5 - very likely
tortilla_calories	guessing calories in a burrito sandwhich from Chipolte? 1 - 580 2 - 725 3 - 940 4 - 1165
turkey_calories	Can you guess how many calories are in the foods shown below? (Panera Bread Roasted Turkey and Avocado BLT) 1 - 345 2 - 500 3 - 690 4 - 850
type_sports	what type of sports are you involved? Open-ended
veggies_day	How likely to eat veggies in a day? 1 - very unlikely 2 - unlikely 3 - neutral

	4- likely 5 - very likely
vitamins	do you take any supplements or vitamins? 1 – yes 2 – no
waffle_calories	guessing calories in waffle potato sandwhich 1 - 575 2 - 760 3 - 900 4 - 1315
weight	what is your weight in pounds?


On branch default

```
[1] LOAD DATASET food_coded AS csv FROM food_coded.csv @ artifact file 56
```

Console Timing Datasets Charts

food_coded (125 rows)

Views

	GPA (string)	Gender (string)	breakfast (string)	calories_chicken (string)	calories_day (string)	calories_scone (string)	coffee (string)	comfort_food (st
0	2.4	2	1	430	nan	315	1	none
1	3.654	1	1	610	3	420	2	chocolate, chips, ice cream
2	3.3	1	1	720	4	420	2	frozen yogurt, pizza, fast food
3	3.2	1	1	430	3	420	2	Pizza, Mac and cheese, ice cream
4	3.5	1	1	720	2	420	2	Ice cream, chocolate, chips
5	2.25	1	1	610	3	980	2	Candy, brownies and soda.
6	3.8	2	1	610	3	420	2	Chocolate, ice cream, french fries, pret
7	3.3	1	1	720	3	420	1	Ice cream, cheeseburgers, chips.
8	3.3	1	1	430	nan	420	1	Donuts, ice cream, chips
9	3.3	1	1	430	3	315	2	Mac and cheese, chocolate, and pasta
10	3.5	1	1	610	3	980	2	Pasta, grandma homemade chocolate
11	3.904	1	1	720	4	420	2	chocolate, pasta, soup, chips, popcorn
12	3.4	2	1	430	3	420	2	Cookies, popcorn, and chips
13	3.6	1	1	610	3	420	2	ice cream, cake, chocolate
14	3.1	2	1	610	3	420	2	Pizza, fruit, spaghetti, chicken and Pota
15	nan	2	2	430	nan	980	2	cookies, donuts, candy bars
16	4	1	1	265	3	420	1	Saltfish, Candy and Kit Kat
17	3.6	2	1	430	3	980	2	chips, cookies, ice cream
18	3.4	1	1	720	3	980	1	Chocolate, ice crea
19	2.2	2	1	430	2	420	2	pizza, wings, Chinese

2. Data Preparation and Cleaning

2.1 Data Preparation problem / solution:

Manual data entry or result of erroneous integration probelem: The food choice questionnaire provides multiple choice questions with numeric types and custom inputs, so when importing vizier for analysis, we chose to use text types for all fields.

2.2 Data Clean problem /solution /challenge:

Problem 1: Unspecified Null Value

We found that people didn't specify what they couldn't answer for some reason, so they filled the table with descriptions that meant something close to Null, such as 'None','nan','unkown'. We need to find these words and cast them to 'Null' in the database instead.

temporary_dataset (5 rows)

Views

	GPA (string)	Gender (string)	breakfast (string)	calories_chicken (string)	calories_day (string)	calories_scone (string)	coffee (string)	comfort_food (string)
0	nan	2	2	430	nan	980	2	cookies, donuts, candy bars
1	Personal	1	1	610	2	980	2	Macaroons, truffles, peanut butter n cho
2	3.79 bitch	2	1	720	4	420	2	Chips, ice cream
3	nan	1	1	610	4	420	2	Pizza, mashed potatoes, spaghetti
4	Unknown	1	1	720	3	420	2	Chips, chocolate, ,mozzarella sticks

Detecting method and handling:

To further detect the existence of this type of record throughout the table, we list roughly the phrase with the meaning of None and find the approximate value using edit distance. Then they are uniformly converted to null in the database.

About the edit distance algo implementation:

[4]

```
# implementation of edit distance algorithm by dynamic programing
def edit_distance(s1, s2):
    m = len(s1)
    n = len(s2)

    # Create a matrix to store the edit distances
    dp = [[0] * (n + 1) for _ in range(m + 1)]

    # Initialize the first row and column
    for i in range(m + 1):
        dp[i][0] = i
    for j in range(n + 1):
        dp[0][j] = j

    # Fill in the rest of the matrix
    for i in range(1, m + 1):
        for j in range(1, n + 1):
            if s1[i - 1] == s2[j - 1]:
                dp[i][j] = dp[i - 1][j - 1]
            else:
                dp[i][j] = min(dp[i - 1][j - 1], dp[i][j - 1], dp[i - 1][j]) + 1

    # Return the edit distance between the two strings
    return dp[m][n]
vizierdb.export_module(edit_distance)
```

Console Timing Datasets Charts

Let's try adjusting the edit distance threshold to 1 and see what the match looks like. We found some correlations with "none" such as "none."," nan","nun".

[5]

```
import math
vizierdb.get_module("edit_distance")
default_ds=vizierdb.get_dataset('food_coded')
nulset=set()
nulset.add("nan")
nulset.add("none")
nulset.add("unknown")
clset = set()
for column in default_ds.columns:
    clset.add(column.name)
app_null_list = []
for row in default_ds.rows:
    for cl in clset:
        v = row.get_value(cl)
        if not v:
            continue
        if v in nulset:
            app_null_list.append("rowid: "+row.identifier+"; ed: 0; value: "+v+"; most like word:"+v)
            row[cl] = None
            continue
        try:
            number = float(v)
        except ValueError:
            v = v.strip()
            for nl in nulset:
                ed = edit_distance(nl,v)
                if ed <2:
                    row[cl]=None
                    app_null_list.append("rowid: "+row.identifier+"; ed: "+str(ed)+"; value: "+v+"; most like word:"+nl)
                    break
for n in app_null_list:
    print(n)
vizierdb.update_dataset('food_coded', default_ds)
```

Detected Result:

```
rowid: 219056892; ed: 0; value: none; most like word:none
rowid: 219056892; ed: 0; value: nan; most like word:nan
rowid: 219056892; ed: 0; value: nan; most like word:nan
rowid: -1598909211; ed: 0; value: nan; most like word:nan
rowid: -1598909211; ed: 0; value: nan; most like word:nan
rowid: -343338021; ed: 0; value: nan; most like word:nan
rowid: 482120217; ed: 0; value: nan; most like word:nan
rowid: 1856144885; ed: 0; value: none; most like word:none
rowid: 1856144885; ed: 0; value: nan; most like word:nan
rowid: -55028412; ed: 0; value: nan; most like word:nan
rowid: -806373830; ed: 0; value: nan; most like word:nan
rowid: -806373830; ed: 0; value: nan; most like word:nan
rowid: 1759326258; ed: 0; value: nan; most like word:nan
rowid: -1838485025; ed: 0; value: none; most like word:none
rowid: 39221538; ed: 0; value: nan; most like word:nan
rowid: 39221538; ed: 0; value: nan; most like word:nan
rowid: 39221538; ed: 0; value: nan; most like word:nan
rowid: 39221538; ed: 0; value: nan; most like word:nan
rowid: 2104643179; ed: 0; value: nan; most like word:nan
rowid: 2104643179; ed: 0; value: nan; most like word:nan
rowid: -1234507861; ed: 0; value: nan; most like word:nan
rowid: -2142889790; ed: 0; value: nan; most like word:nan
rowid: 1952942318; ed: 0; value: nan; most like word:nan
rowid: 605970900; ed: 0; value: nan; most like word:nan
rowid: 605970900; ed: 0; value: nan; most like word:nan
rowid: -1429761528; ed: 0; value: nan; most like word:nan
rowid: -1429761528; ed: 0; value: nan; most like word:nan
rowid: -1429761528; ed: 1; value: Unknown; most like word:unknown
rowid: -1429761528; ed: 0; value: nan; most like word:nan
rowid: -1429761528; ed: 0; value: nan; most like word:nan
rowid: 1695531960; ed: 0; value: none; most like word:none
rowid: 1695531960; ed: 0; value: nan; most like word:nan
```

Then converted them to null in the database. Chose one column to check if the method works out:

[6]

```
select * from food_coded where calories_day = 'nan'
```

Console Timing Datasets Charts

temporary_dataset (0 rows)

Views

GPA (string)	Gender (string)	breakfast (string)	calories_chicken (string)	calories_day (string)	calories_scone (string)	coffee (string)	comfort_food (string)	comfort_food_re

Challenge:

After a large number of nulls are unified, we find that there are still some expressions in the table that are close to null like the column weight has the value called 'I'm not answering this.'.Traditional algorithms for string approximation are unable to recognize human language, and we think we can recognize such records by using some machine learning methods for natural language recognition.

[7]

```
select weight from food_coded where weight not regexp '^[0-9]+(\.[0-9]+)?$'
```

Console Timing Datasets Charts

temporary_dataset (3 rows)

Views

	weight (string)
0	I'm not answering this.
1	Not sure, 240
2	144 lbs

Problem 2: Switching Fields

In order to further detect the data problem caused by the type, we separately find all the fields in the whole table that should be numbers, and if the records under these fields are not numbers, they are printed out in the console:

[6]

```
import re
default_ds=vizierdb.get_dataset('food_coded')
non_numeric_set=set()
non_numeric_set.add('comfort_food')
non_numeric_set.add('comfort_food_reasons')
non_numeric_set.add('diet_current')
non_numeric_set.add('father_profession')
non_numeric_set.add('fav_cuisine')
non_numeric_set.add('food_childhood')
non_numeric_set.add('healthy_meal')
non_numeric_set.add('ideal_diet')
non_numeric_set.add('meals_dinner_friend')
non_numeric_set.add('mother_profession')
non_numeric_set.add('type_sports')
non_numeric_set.add('eating_changes')


clset = set()
for column in default_ds.columns:
    clset.add(column.name)

numeric_set = clset - non_numeric_set
for row in default_ds.rows:
    for cl in numeric_set:
        v=row.get_value(cl)
        if not v:
            continue
        v = v.strip()
        try:
            number = float(v)
        except ValueError:
            print(row.identifier,"->",cl,"->",v)
#vizierdb.update_dataset('food_coded', default_ds)
```

Console Timing Datasets Charts

Interestingly, we found a strange record which most of the field types were wrong. After analysis, it was found that vizier used commas as delimiters when importing data and divided a line of records in one field into multiple copies, incorrectly occupying other fields. Here, we use row.identifier(-438992023) to locate this problem.

Console ▾ Timing Datasets ▾ Charts ▾

```
-1518051663 -> weight -> I'm not answering this.
184206148 -> weight -> Not sure, 240
-438992023 -> cook -> and anxiety "
-438992023 -> ideal_diet_coded -> chicken, veggies, fruit, water
-438992023 -> income -> My ideal diet would be to add more fruits and veggies. Also, to go to the gym more but school gets i
-438992023 -> drink -> My diet is pretty healthy and consists of chicken, turcky, and veggies. I rarely eat unhealthy food o
-438992023 -> fav_cuisine_coded -> Dead beat
-438992023 -> comfort_food_reasons_coded1 -> special occasions
-438992023 -> GPA -> Personal
-438992023 -> fav_food -> Mexican
-438992023 -> fruit_day -> Chocolate
-438992023 -> on_off_campus -> Respiratory Therapist
-438992023 -> eating_changes_coded2 -> I have gotten healthier with my eating habits. As a result my parents starting eating
677615185 -> weight -> 144 lbs
375855847 -> GPA -> 3.79 bitch
570572914 -> GPA -> Unknown
```

peanut butter, dessets, pretzels.	Sadness, boredom, lonely.	3	2	3
Macaroons, truffles, peanut butter n chocolate ice cream	"I do not really eat ""comfort food"" but I guess sadness	special occasions	and anxiety "	3
ice cream, cookies, ice cream	boredom, sadness	2	3	2

Problem 3: Usless Information

For a purely numeric field, we should keep the numeric result and remove the useless text description.For example, 'Personal' '3.79 bitch' is found in GPA fields, as shown in the following figure.

[8]

```
select gpa from food_coded where gpa not regexp '^[0-9]+(\.[0-9]+)?$'
```

Console ▾ Timing Datasets ▾ Charts ▾

temporary_dataset (2 rows)

Views

	gpa (string)
0	Personal
1	3.79 bitch

Detecting method and handling:

We find purely numeric fields and use regular expressions to find the wrong types. Extracting the longest number from it.

Console Timing Datasets Charts

```
rowid:-1518051663, column:weight, value:I'm not answering this.; After extracting:
rowid:184206148, column:weight, value:Not sure, 240; After extracting:240
rowid:-2073641407, column:GPA, value:Personal; After extracting:
rowid:677615185, column:weight, value:144 lbs; After extracting:144
rowid:375855847, column:GPA, value:3.79 bitch; After extracting:3.79
```

Challenge: Similar to the challenge of null processing, we can not extract results entirely from expressions and can not solve all problems.

The longest number algorithm:

[9]

```python
import re
default_ds=vizierdb.get_dataset('food_coded')

# extract the longest number
def extract_longest_number_string(input_string):
    current_number = ""
    longest_number = ""
    for char in input_string:
        if char.isdigit() or char == '.':
            current_number += char
        else:
            if len(current_number) > len(longest_number):
                longest_number = current_number
            current_number = ""
    if len(current_number) > len(longest_number):
        longest_number = current_number
    if longest_number == '.':
        return ""
    return longest_number
```

Detecting and handling:

```python
#define the non numeric columns
non_numeric_set=set()
non_numeric_set.add('comfort_food')
non_numeric_set.add('comfort_food_reasons')
non_numeric_set.add('diet_current')
non_numeric_set.add('father_profession')
non_numeric_set.add('fav_cuisine')
non_numeric_set.add('food_childhood')
non_numeric_set.add('healthy_meal')
non_numeric_set.add('ideal_diet')
non_numeric_set.add('meals_dinner_friend')
non_numeric_set.add('mother_profession')
non_numeric_set.add('type_sports')
non_numeric_set.add('eating_changes')
clset = set()
for column in default_ds.columns:
    clset.add(column.name)
numeric_set = clset - non_numeric_set

str_list=[]
for row in default_ds.rows:
    for cl in numeric_set:
        v = row.get_value(cl)
        if not v:
            continue
        if not re.match('^[0-9]+(\.[0-9]+)?$',v):
            hlstr = extract_longest_number_string(v)
            str_list.append("rowid:"+row.identifier + "," +" column:"+cl + ", value:"+v+"; After extracting:"+hlstr)
            if not hlstr:
                hlstr = None
            row[cl]=hlstr
for n in str_list:
    print(n)
vizierdb.update_dataset('food_coded', default_ds)
```

After updating the bad records:

[10]

```
select * from food_coded where gpa not regexp '^[0-9]+(\.[0-9]+)?$'
```

Console | Timing | Datasets | Charts

temporary_dataset (0 rows)

Views

GPA (string)	Gender (string)	breakfast (string)	calories_chicken (string)	calories_day (string)	calories_scone (string)	coffee (string)	comfort_food (string)	comfort_food_r

2.2 Constraint-based cleaning

First work focused on functional dependencies

FD1: comfort food reason (i.e., anger, sadness, happiness, boredom, etc- list up to three) **->** comfort food reason coded(1 – stress 2 – boredom ,3 – depression/sadness ,4 – hunger, 5 – laziness ,6 – cold weather,7 – happiness ,8-watching tv,9 – none)

Records: **Stress**, bored, ange -> **1 (stress)**

If the situation described in the comfort food reason is one of the first eight options in the comfort food reason coded, they have a correspondence, and the other cases correspond to option 9.

comfort food reason coded relies on descriptions of comfort food reasons, but there is little we can do to draw definitive conclusions from a single sentence. For example, when the comfort food reason states 'A long day, not feeling well, winter ', the program cannot infer that the statement represents 'depression' or 'cold weather'. We might be able to use natural language processing to try to infer conclusions.

Here we use the edit distance and the longest string algorithms to try to match:

[13]

```
default_ds=vizierdb.get_dataset('food_coded')
vizierdb.get_module("extract_longest_number_string")
vizierdb.get_module("edit_distance")
dectect_cl = set()
dectect_cl.add('comfort_food_reasons')
dectect_cl.add('comfort_food_reasons_coded1')

coded_set_dict={}
coded_set_dict['stress']=1
coded_set_dict['boredom']=2
coded_set_dict['depression']= 3
coded_set_dict['sadness']= 3
coded_set_dict['hunger']= 4
coded_set_dict['laziness']=5
coded_set_dict['cold weather']=6
coded_set_dict['happiness']=7
coded_set_dict['watching tv']=8

for row in default_ds.rows:
    c_f_r = row.get_value('comfort_food_reasons')
    c_f_r_c = row.get_value('comfort_food_reasons_coded1')
    if not c_f_r :
        continue
    vlist = c_f_r.split(",")
    is_predicted = False
    for v in vlist:
        ed = 9999
        predict_v = ""
        for key,value in coded_set_dict.items():
            ed_actual = edit_distance(v,key)
            if ed_actual < ed:
                ed = ed_actual
                predict_v = key
        predict_option = coded_set_dict[predict_v]
        actual_option = c_f_r_c
        print("rowid:"+str(row.identifier)+"; actual - cf reason :"+str(c_f_r)+"; actual - cf reason coded :"+str(actua
        if str(actual_option).strip() == str(predict_option).strip():
            is_predicted = True
            break
    print("Predict Hit:"+str(is_predicted) +"; rowid:"+str(row.identifier))
```

```
rowid:1412132707; actual - cf reason :we dont have comfort; actual - cf reason coded :9; predict reason: boredom; predict coded
Predict Hit:False; rowid:1412132707
rowid:2049010962; actual - cf reason :Stress, bored, anger; actual - cf reason coded :1; predict reason: stress; predict coded:
Predict Hit:True; rowid:2049010962
rowid:-1518051663; actual - cf reason :stress, sadness; actual - cf reason coded :1; predict reason: stress; predict coded:1
Predict Hit:True; rowid:-1518051663
rowid:184206148; actual - cf reason :Boredom; actual - cf reason coded :2; predict reason: boredom; predict coded:2
Predict Hit:True; rowid:184206148
rowid:2078747405; actual - cf reason :Stress, boredom, cravings ; actual - cf reason coded :1; predict reason: stress; predict
Predict Hit:True; rowid:2078747405
rowid:386996704; actual - cf reason :None, i don't eat comfort food. I just eat when i'm hungry.; actual - cf reason coded :4;
Predict Hit:True; rowid:386996704
rowid:-671897412; actual - cf reason :stress, boredom; actual - cf reason coded :1; predict reason: stress; predict coded:1
Predict Hit:True; rowid:-671897412
rowid:2051450779; actual - cf reason :I eat comfort food when im stressed out from school(finals week), when I`m sad, or when i
rowid:2051450779; actual - cf reason :I eat comfort food when im stressed out from school(finals week), when I`m sad, or when i
Predict Hit:True; rowid:2051450779
rowid:-1587011751; actual - cf reason :Boredom; actual - cf reason coded :2; predict reason: boredom; predict coded:2
Predict Hit:True; rowid:-1587011751
rowid:-1021083878; actual - cf reason :Stress, anger and sadness ; actual - cf reason coded :1; predict reason: stress; predict
Predict Hit:True; rowid:-1021083878
rowid:806337751; actual - cf reason :Boredom; actual - cf reason coded :2; predict reason: boredom; predict coded:2
Predict Hit:True; rowid:806337751
rowid:-792686031; actual - cf reason :sadness, stress, cold weather; actual - cf reason coded :3; predict reason: sadness; pred
Predict Hit:True; rowid:-792686031
rowid:1665562717; actual - cf reason :Sadness, boredom, late night snack ; actual - cf reason coded :3; predict reason: sadness
Predict Hit:True; rowid:1665562717
rowid:-1598281284; actual - cf reason :stress,  boredom, special occasions; actual - cf reason coded :1; predict reason: stress
Predict Hit:True; rowid:-1598281284
rowid:-317265190; actual - cf reason :Friends, environment and boredom; actual - cf reason coded :2; predict reason: stress; pr
rowid:-317265190; actual - cf reason :Friends, environment and boredom; actual - cf reason coded :2; predict reason: boredom; p
Predict Hit:True; rowid:-317265190
```

Challenge:

To infer a word from a sentence using natural language processing (NLP) techniques in Python, we can leverage language models and probabilistic inference. One common approach is based on probability estimation using pre-trained language models such as GPT-2 or BERT. By calculating the probabilities of each possible word given the context, we can select the word with the highest probability as the inferred result.

Denial Constrains

In most fields, we provide explicit options and values, and values outside of these options should not occur.If there are other options that represent "none" or indicate the absence of a value, you can handle them by assigning a specific value to represent that.

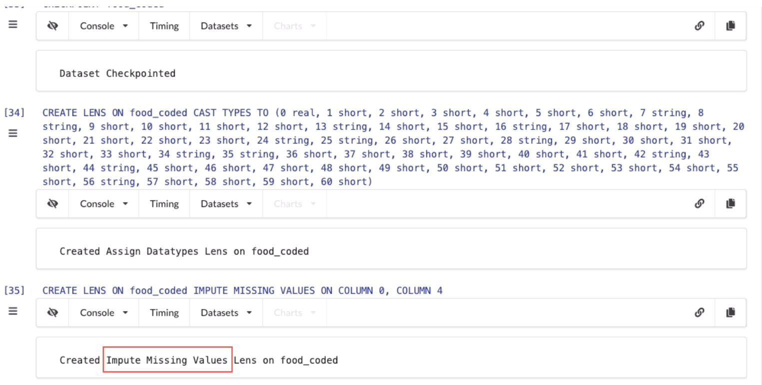
[30]

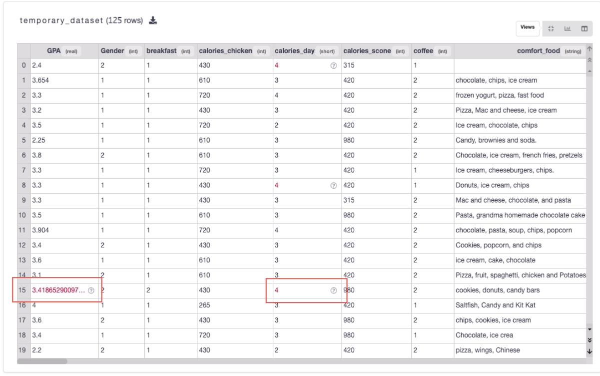
```
default_ds=vizierdb.get_dataset('food_coded')

fill_set_dict={}
fill_set_dict['comfort_food_reasons_coded1']='9'
fill_set_dict['cook']='5'
fill_set_dict['cuisine']= '6'
fill_set_dict['diet_current_coded']= '4'
fill_set_dict['eating_changes_coded1']= '4'
fill_set_dict['employment']='4'
fill_set_dict['ideal_diet_coded']='8'


for row in default_ds.rows:
    for key,value in fill_set_dict.items():
        value = row[key]
        if not value:
            row[key]=fill_set_dict.get(key)
            print("rowid:"+str(row.identifier)+"; column:"+key+"; cover by:"+fill_set_dict.get(key))
vizierdb.update_dataset('food_coded', default_ds)
```

Console | Timing | Datasets | Charts

```
rowid:1412132707; column:cuisine; cover by:6
rowid:386996704; column:cuisine; cover by:6
rowid:-1598909211; column:cuisine; cover by:6
rowid:-806373830; column:cuisine; cover by:6
rowid:39221538; column:cuisine; cover by:6
rowid:-1234507861; column:cuisine; cover by:6
rowid:-1429761528; column:employment; cover by:4
rowid:-1786727130; column:employment; cover by:4
rowid:1779387099; column:employment; cover by:4
rowid:-57910806; column:employment; cover by:4
rowid:23501288; column:employment; cover by:4
rowid:636965049; column:cuisine; cover by:6
rowid:-2073641407; column:cuisine; cover by:6
rowid:1740455480; column:cuisine; cover by:6
rowid:1558341968; column:cook; cover by:5
rowid:1040215375; column:employment; cover by:4
rowid:505761586; column:cook; cover by:5
rowid:-114872478; column:cuisine; cover by:6
```

After that, we use vizier field guesses to convert the original string type to what it should be, and use the impute missing value feature to fill in the missing values.





temporary_dataset (125 rows)

	GPA (real)	Gender (int)	breakfast (int)	calories_chicken (int)	calories_day (short)	calories_scone (int)	coffee (int)	comfort_food (string)
0	2.4	2	1	430	4	315	1	
1	3.654	1	1	610	3	420	2	chocolate, chips, ice cream
2	3.3	1	1	720	4	420	2	frozen yogurt, pizza, fast food
3	3.2	1	1	430	3	420	2	Pizza, Mac and cheese, ice cream
4	3.5	1	1	720	2	420	2	Ice cream, chocolate, chips
5	2.25	1	1	610	3	980	2	Candy, brownies and soda.
6	3.8	2	1	610	3	420	2	Chocolate, ice cream, french fries, pretzels
7	3.3	1	1	720	3	420	1	Ice cream, cheeseburgers, chips.
8	3.3	1	1	430	4	420	1	Donuts, ice cream, chips
9	3.3	1	1	430	3	315	2	Mac and cheese, chocolate, and pasta
10	3.5	1	1	610	3	980	2	Pasta, grandma homemade chocolate cake
11	3.904	1	1	720	4	420	2	chocolate, pasta, soup, chips, popcorn
12	3.4	2	1	430	3	420	2	Cookies, popcorn, and chips
13	3.6	1	1	610	3	420	2	ice cream, cake, chocolate
14	3.1	2	1	610	3	420	2	Pizza, fruit, spaghetti, chicken and Potatoes
15	3.41865290097...	2	2	430	4	980	2	cookies, donuts, candy bars
16	4	1	1	265	3	420	1	Saltfish, Candy and Kit Kat
17	3.6	2	1	430	3	980	2	chips, cookies, ice cream
18	3.4	1	1	720	3	980	1	Chocolate, ice crea
19	2.2	2	1	430	2	420	2	pizza, wings, Chinese

Source code and original dataset can be found on github:

https://github.com/IITTeaching/cs520-f23-group-5/tree/main/data_curation_project