APA - Data Preparation Template

DATA 3300

Name(s):

Read and follow these assignment instructions carefully! Ordinarily, we'd jump into cleansing data as needed and we'd each do it slightly differently. This is difficult to grade, so please go in the order of this document and follow these instructions.

For this assignment students should submit one .ipynb file, one clean, sorted Excel file, and a PDF of this iPython notebook file. See the assignment background for attribute descriptions and the assignment requirements link on the Canvas assignment page.

Students should alter code in lines with "replace code..." language. Otherwise, students do not need to alter the provided code.

Q1

What does the term "data quality" refer to and why is it important within the context of using data to solve business problems?

The degree to which a dataset can be efficiently and effectively processed and used. It is important because it helps create analyses that are meaningful, trustworthy, and that do not induce mayhem.

Q2

The lecture and textbook discussed five characteristics of data quality: accuracy, completeness, consistency, timeliness, and uniqueness. While "accuracy" in this dataset is hard for us to gauge without further inquiry and"timeliness" kind of depends on what it is the data are to be used for, there are clear examples of problems with each of the other three characteristics.

For each of the other three (completeness, consistency, uniqueness), identify a specific situation within the *hbdata-orig.csv* dataset (identify record numbers when applicable). Hint: For uniqueness, look at the VisitSpan attribute. Be sure your answers are clearly labelled and described.

This can be done by viewing the data in Excel or importing it into Python. If using Python, begin by importing your libraries and then read in the .csv file.

```
import pandas as pd
import numpy as np

#just run this code block after importing libraries above, no need to alter the below code!
from datetime import datetime #you'll also need this method to convert a variable to date-time later in the assignment
import warnings
warnings.filterwarnings("ignore") #this can be removed but will help the code be a little cleaner by ignoring warnings


#replace with code for importing the dataset
df = pd.read_excel('/content/hbdata-orig24.xlsx')
#run this code block after importing dataset above
pd.set_option("display.max_rows", None, "display.max_columns", None) #View full dataframe
df
```

	VisitID	BirdName	BirdType	BirdSize	Gender	VisitDate	VisitSpan	Locat
0	2203	Rose Kim	Rufus	S	female	2017-08-03 00:00:00	23:13:05.736-23:13:21.109	Back I
1	2204	Horace Prefect	Calliope	M	male	2017-07-16 00:00:00	07:45:07.65-07:45:17.97	NE Cc
2	2205	Edeltraud McKnight	Rufous	M	female	2017-07-16 00:00:00	08:09:52.10-08:10:22.74	De
3	2206	Herbert McKnight	Rufous	M	male	2017-07-16 00:00:00	08:41:54.07-08:42:07.48	De
4	2207	Walpurga Schwalbe	Anna's	M	female	2017-07-16 00:00:00	09:18:59.33-09:19:13.41	Ga
5	2208	Walpurga Schwalbe	Anna's	M	female	2017-07-16 00:00:00	09:28:41.64-09:28:52.64	De
6	2209	NaN	NaN	NaN	NaN	2017-07-16 00:00:00	10:56:26.40-10:56:45.32	De
7	2210	Blossom Kim	Rufous	M	female	2017-07-16 00:00:00	11:27:05.53-11:27:31.41	De
8	2211	Gale Sharma	Rufous	XL	female	2017-07-16 00:00:00	12:14:11.70-12:14:35.72	Back I
9	2212	King Schwalbe	Anna's	L	male	2017-07-16 00:00:00	12:14:45.98-12:15:14.45	NE Cc
10	2213	Lucky Kim	Rufous	M	male	2017-07-16 00:00:00	13:07:55.82-13:08:11.09	De
11	2214	Ruby Schwalbe	Anna's	S	female	2017-07-16 00:00:00	14:08:35.90-14:08:41.95	De
12	2215	Norm Sharma	Rufous	M	male	2017-07-16 00:00:00	15:15:15.55-15:15:35.11	Back I
13	2216	Petunia O'Flaherty	Anna's	S	female	2017-07-16 00:00:00	15:38:48.39-15:39:11.49	De
14	2217	Sol Schwalbe	Anna's	M	male	2017-07-16 00:00:00	16:58:39.17-16:58:57.62	Back I
15	2218	Mitz Johnson	Rufus	M	female	2017-07-16 00:00:00	17:25:12.74-17:25:34.44	De
16	2219	Francine Johnson	Rufous	M	female	2017-07-17 00:00:00	07:29:33.89-07:29:56.56	De
17	2220	June McKnight	Rufous	M	female	2017-07-17 00:00:00	09:52:51.78-09:53:14.73	Back I
18	2221	Lucky Kim	Rufous	M	male	2017-07-17 00:00:00	10:25:08.10-10:25:29.68	Back I
		King				2017-07-17	11:09:34.78-	

- Completeness: VisitIDs 2209, 2395, missing data
- Consistency: VisitIDs 2250-2257, medium instead of M. Lots of oz instead of ml as well
- Uniqueness: 2235-2336: Time is the same, duplicate entry

Q3

Answer the following questions regarding the importance of each of the named data scrubbing steps.

3A

What negative outcome would likely occur during data analysis if duplicated visits were not removed (consolidated)?

The dulicate data would be recorded multiple times, scewing the results of a particular bird or feeder. We would likely find that there are more birds observed than there should be, and their results would skew what we assume about that bird.

3B

What negative outcome would likely occur during data analysis if Location entries were not standardized?

Some locations would have missing entries, for example "Deck. 2" would not be counted in "Deck 2"

3C

What negative outcome would likely occur during data analysis if, due to measurement issues, amount eaten values were not consistent?

One might assume a bird has eaten much more or much less than it actually has. This can cause future predictions of how much to feed to be wrong.

Q4

In scrubbing the data, you should address each of the following. It's recommended to go through this assignment in the order of steps below.

4A

In Python trim all text columns to ensure there are no leading or trailing spaces on any of the data values. If you still see two seemingly identical entries, remember that trimming does not trim internal spaces between two words.

```
df_obj = df.select_dtypes(['object']) #select all columns containing text (of type object)
df[df_obj.columns] = df_obj.apply(lambda x: x.str.strip()) #trim all selected columns
#replace with code to view the header of the dataframe (cheat sheet!)
df.head()
#keep below code to verify date/time of your work
from datetime import datetime
now = datetime.now()
formatted_time = now.strftime("%Y-%m-%d %H:%M:%S")
print(f"Current Date and Time: {formatted_time}")
```

```
Current Date and Time: 2024-01-23 20:32:31
```

4B

Ensure each bird's first and last name should be included in separate fields (columns).

```
#replace with code to split column, updating column names and delimiter type (cheat sheet)
df[['BirdFirstName','BirdLastName']] = df['BirdName'].str.split(' ', expand=True)
#replace with code to drop birdname column
df = df.drop('BirdName', axis = 1)
#replace with code to show df heading
df.head()
```

	VisitID	BirdType	BirdSize	Gender	VisitDate	VisitSpan	Location	Hoverfeed
0	2203	Rufus	S	female	NaN	23:13:05.736-23:13:21.109	Back Patio	yes
1	2204	Calliope	M	male	NaN	07:45:07.65-07:45:17.97	NE Corner	yes
2	2205	Rufous	M	female	NaN	08:09:52.10-08:10:22.74	Deck 2	no

4C

Ensure that data values are standardized in each column (pay particular attention to the textual columns, BirdName, BirdType, BirdSize, Gender, Location, and Hoverfeed) so that each "idea" (e.g., a species of bird) is specified (i.e., spelled) the same way.

This can be checked for each column by using the .value_counts() method! After applying the standarization, show that the correction has been made by using the .value_counts() method to display all categories for each column of interest.

```
#Replace with code to examine levels in BirdType using the value_counts() method, update column name
df['BirdType'].value_counts()
```

```
Rufous      103
Anna's       85
Calliope     61
Rufus        18
Prefect      14
Name: BirdType, dtype: int64
```

```
#Replace with code to replace redundant value in BirdType, for only 1 original and new label
df = df.replace('Rufus','Rufous')
```

```
#Replace with code to examine levels of BirdSize. Ignore null, "medium" and 20ml lines; these will be fixed later.
df['BirdSize'].value_counts()
```

```
M         162
S          48
L          33
XL         23
medium     13
20 ml       5
Name: BirdSize, dtype: int64
```

```
#Replace with code to replace redundant value in BirdSize
df = df.replace('medium','M')
```

```
#Replace with code to examine levels of Location. Ignore two lines with all numerical output; these will be fixed later.
df['Location'].value_counts()
```

```
Back Patio                68
Deck 2                    54
Deck 1                    46
NE Corner                 44
Garage                    29
Entry                     29
Back  Patio                8
Deck. 2                    7
15:16:40.19-15:16:55.98    4
13:14:33.36-13:15:09.25    2
Name: Location, dtype: int64
```

```
#Replace with code to replace redundant values of Location, for 2 original and new labels
df = df.replace({'Back  Patio':'Back Patio','Deck. 2':'Deck 2'})
```

4D

The amount eaten is very important to our analysis. Where this value is null (blank), omit the record. Note that values of 0 are not the same as null. Retain (don't delete) the records where the value of Amount is 0. Other issues with this attribute will be addressed later.

Before removing missing columns first examine the length of the dataset currently using the len() method. Then split the *Amount* column into two columns, *Amount_Eat* and *Unit*. Finally, change *Amount_Eat* to a float object using pd.to_numeric().

Once missing values are dropped, view the length of the dataset again using the len() method to see how many observations were dropped.

```
len(df) #return length of dataframe (i.e., number of rows), just run this code block
```

```
291
```

```
#Replace with code to create new split columns for 'Amount_Eat' and 'Unit' from the 'Amount' column, updating column names and d
df[['Amount_Eat','Unit']] = df['Amount'].str.split(' ', expand=True)
#Replace with code to drop 'Amount' column
df = df.drop('Amount', axis = 1)
#Replace with code to show header of dataframe
df.head()
```

	VisitID	BirdType	BirdSize	Gender	VisitDate	VisitSpan	Location	Hoverfeed
0	2203	Rufous	S	female	NaN	23:13:05.736-23:13:21.109	Back Patio	yes
1	2204	Calliope	M	male	NaN	07:45:07.65-07:45:17.97	NE Corner	yes
2	2205	Rufous	M	female	NaN	08:09:52.10-08:10:22.74	Deck 2	no

```
df['Amount_Eat'] = df['Amount_Eat'].apply(pd.to_numeric, errors='coerce') #convert Amount_Eat column variable dtype from object
#just run this code block
df
```

	VisitID	BirdType	BirdSize	Gender	VisitDate	VisitSpan	Location	Hoverf
0	2203	Rufous	S	female	NaN	23:13:05.736-23:13:21.109	Back Patio	
1	2204	Calliope	M	male	NaN	07:45:07.65-07:45:17.97	NE Corner	
2	2205	Rufous	M	female	NaN	08:09:52.10-08:10:22.74	Deck 2	
3	2206	Rufous	M	male	NaN	08:41:54.07-08:42:07.48	Deck 1	
4	2207	Anna's	M	female	NaN	09:18:59.33-09:19:13.41	Garage	
5	2208	Anna's	M	female	NaN	09:28:41.64-09:28:52.64	Deck 2	
6	2209	NaN	NaN	NaN	NaN	10:56:26.40-10:56:45.32	Deck 1	
7	2210	Rufous	M	female	NaN	11:27:05.53-11:27:31.41	Deck 1	
8	2211	Rufous	XL	female	NaN	12:14:11.70-12:14:35.72	Back Patio	
9	2212	Anna's	L	male	NaN	12:14:45.98-12:15:14.45	NE Corner	
10	2213	Rufous	M	male	NaN	13:07:55.82-13:08:11.09	Deck 2	
11	2214	Anna's	S	female	NaN	14:08:35.90-14:08:41.95	Deck 2	
12	2215	Rufous	M	male	NaN	15:15:15.55-15:15:35.11	Back Patio	
13	2216	Anna's	S	female	NaN	15:38:48.39-15:39:11.49	Deck 2	
14	2217	Anna's	M	male	NaN	16:58:39.17-16:58:57.62	Back Patio	
15	2218	Rufous	M	female	NaN	17:25:12.74-17:25:34.44	Deck 2	
16	2219	Rufous	M	female	NaN	07:29:33.89-07:29:56.56	Deck 1	
17	2220	Rufous	M	female	NaN	09:52:51.78-09:53:14.73	Back Patio	
18	2221	Rufous	M	male	NaN	10:25:08.10-10:25:29.68	Back Patio	
						11:09:34.78-		