DSC 550 Final Project - Deborah Young

June 1, 2023

0.0.1 Introduction

Research around the human microbiome (the collection of bacteria and microbes that live in and on our bodies) has been growing as a science and field of interest more and more over the last decade (Ursell et al., 2012). Many people experience disruptions to their health that could be attributed to an imbalance of these little organisms that share life with each of us (Ursell et al., 2012). Gastrointestinal (GI/gut) health is of particular interest, and investigations around how to improve this ecosystem within us are expanding. One approach to achieve this is Fecal Microbiota Transplant (FMT).

FMT is a method of transferring a healthy population (high species quantity and diversity) of microbes from one person to another in hopes of restoring the bacterial colonies in the recipient and thus contributing to greater overall health (Orr, et al. 2018). FMT has been used as a therapy for a limited selection of illnesses (and is not approved for many uses in the USA), but it is gaining relevance as a therapeutic application for a wide variety of infections, gastrointestinal issues, and auto-immune disorders (Gupta et al., 2016). FMT could be an effective intervention for a variety of health issues caused by diet, environment, overuse of antibiotics, and other factors.

In order to successfully implement an intervention such as this, we need to understand the structure of human microbiota. Using data sourced from NIH Human Microbiome Project, I will run an EDA to explore the species types and prevalence. For the EDA portion, I will be roughly following this Kaggle notebook. The dataset utilized for that project is older (as the project is in constant motion), so although it is a great framework, my approach will have to be adjusted. Moving beyond this step, I would like to build models to predict the presence of these microbes to potentially I will likely reference this or this dataset.

According to AWS: "The NIH-funded Human Microbiome Project (HMP) is a collaborative effort of over 300 scientists from more than 80 organizations to comprehensively characterize the microbial communities inhabiting the human body and elucidate their role in human health and disease. To accomplish this task, microbial community samples were isolated from a cohort of 300 healthy adult human subjects at 18 specific sites within five regions of the body (oral cavity, airways, urogenital track, skin, and gut). Targeted sequencing of the 16S bacterial marker gene and/or whole metagenome shotgun sequencing was performed for thousands of these samples. In addition, whole genome sequences were generated for isolate strains collected from human body sites to act as reference organisms for analysis. Finally, 16S marker and whole metagenome sequencing was also done on additional samples from people suffering from several disease conditions."

- Gupta, S., Allen-Vercoe, E., & Petrof, E. O. (2016). Fecal microbiota transplantation: in perspective. Therapeutic Advances in Gastroenterology, 9(2), 229-239. https://doi.org/10.1177/1756283X15607414

- Orr, M. R., Kocurek, K. M., & Young, D. L. **(that's me!)** (2018). Gut Microbiota and Human Health: Insights From Ecological Restoration. The Quarterly Review of Biology, 93(2), 73–90. https://doi.org/10.1086/698021
- Ursell, L. K., Metcalf, J. L., Parfrey, L. W., & Knight, R. (2012). Defining the Human Microbiome. Nutrition reviews, 70(Suppl 1), S38. https://doi.org/10.1111/j.1753-4887.2012.00493.

Milestone 1

0.0.2 Import and Cleaning

[9]:
```
#import libraries and dataset
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')

#part 2
from sklearn import metrics
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import StandardScaler, MinMaxScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression

from sklearn.pipeline import Pipeline, FeatureUnion

microbes=pd.read_csv('/Users/debane/Documents/MS Data Science/550 Data Mining/
 ↪project_catalog.csv')
#####pd.set_option('display.max_rows', None, 'display.max_columns', None)
```

[10]:
```
#View sample of data
microbes.head()
```

[10]:
```
   HMP ID  GOLD ID                        Organism Name     Domain
0       1  Gi03551     Abiotrophia defectiva ATCC 49176  BACTERIAL  \
1       4  Gi03555  Achromobacter piechaudii ATCC 43553  BACTERIAL
2       5  Gi03554       Achromobacter xylosoxidans C54  BACTERIAL
3      10  Gi03422   Acinetobacter baumannii ATCC 19606  BACTERIAL
4      12  Gi03421  Acinetobacter calcoaceticus RUH2202  BACTERIAL

  NCBI Superkingdom HMP Isolation Body Site Project Status
0          Bacteria                    oral       Complete  \
1          Bacteria                airways       Complete
2          Bacteria                airways       Complete
3          Bacteria        urogenital_tract       Complete
```

```
4          Bacteria                    skin       Complete

                Current Finishing Level
0  Level 3: Improved-High-Quality Draft  \
1           Level 2: High-Quality Draft
2      Level 5: Non-contiguous Finished
3           Level 2: High-Quality Draft
4           Level 2: High-Quality Draft

                            NCBI Submission Status  NCBI Project ID
0  6. annotation (and sequence) public on NCBI site            33011  \
1  6. annotation (and sequence) public on NCBI site            46343
2  6. annotation (and sequence) public on NCBI site            38739
3  6. annotation (and sequence) public on NCBI site            38509
4  6. annotation (and sequence) public on NCBI site            38337

     Genbank ID  Gene Count  IMG/HMP ID        HOMD ID
0  ACIN00000000        1950   643886181  HOMD: tax_389  \
1  ADMS00000000        5755   647000200            NaN
2  ACRC00000000        6010           0  HOMD: tax_343
3  ACQB00000000        3832   647533101  HOMD: tax_554
4  ACPK00000000        3632   646206267            NaN

                              Sequencing Center
0  Washington University Genome Sequencing Center  \
1                     Baylor College of Medicine
2                                Broad Institute
3                                Broad Institute
4                                Broad Institute

                Funding Source             Strain Repository ID  Unnamed: 17
0  NIH-HMP Jumpstart Supplement           ATCC 49176, CIP 103242          NaN  \
1  NIH-HMP Jumpstart Supplement  ATCC 43553, CIP 55774, LMG 6100          NaN
2  NIH-HMP Jumpstart Supplement                      BEI HM-235          NaN
3  NIH-HMP Jumpstart Supplement             ATCC 19606, DSM 6974          NaN
4  NIH-HMP Jumpstart Supplement                         LMG 10517          NaN

   Unnamed: 18
0          NaN
1          NaN
2          NaN
3          NaN
4          NaN
```

[11]:
```
#Check shape
microbes.shape
```

[11]: (2915, 19)

[12]:
```
#See column names in dataset
microbes.columns
```

[12]:
```
Index(['HMP ID', 'GOLD ID', 'Organism Name', 'Domain', 'NCBI Superkingdom',
       'HMP Isolation Body Site', 'Project Status', 'Current Finishing Level',
       'NCBI Submission Status', 'NCBI Project ID', 'Genbank ID', 'Gene Count',
       'IMG/HMP ID', 'HOMD ID', 'Sequencing Center', 'Funding Source',
       'Strain Repository ID', 'Unnamed: 17', 'Unnamed: 18'],
      dtype='object')
```

[13]:
```
#View descriptions of data
microbes.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2915 entries, 0 to 2914
Data columns (total 19 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   HMP ID                   2915 non-null   int64
 1   GOLD ID                  1783 non-null   object
 2   Organism Name            2915 non-null   object
 3   Domain                   2712 non-null   object
 4   NCBI Superkingdom        2751 non-null   object
 5   HMP Isolation Body Site  2915 non-null   object
 6   Project Status           2915 non-null   object
 7   Current Finishing Level  1579 non-null   object
 8   NCBI Submission Status   2915 non-null   object
 9   NCBI Project ID          2915 non-null   int64
 10  Genbank ID               1579 non-null   object
 11  Gene Count               2915 non-null   int64
 12  IMG/HMP ID               2915 non-null   int64
 13  HOMD ID                  397 non-null    object
 14  Sequencing Center        2911 non-null   object
 15  Funding Source           2915 non-null   object
 16  Strain Repository ID     1377 non-null   object
 17  Unnamed: 17              0 non-null      float64
 18  Unnamed: 18              0 non-null      float64
dtypes: float64(2), int64(4), object(13)
memory usage: 432.8+ KB
```

There is a broad range of information here, some of which may be beneficial to study regardless of project status, but for efficacy of this project, I'd like to check how many of the entires are complete.

[14]:
```
microbes['Project Status'].value_counts()
```

[14]:
```
Project Status
Complete       1579
In Progress    1336
Name: count, dtype: int64
```

I'm going to remove any entries that are "in progress" from the main dataframe and place them in a new dataframe so I have it for running later if I want.

[15]:
```
# Split Dataframe using groupby() &
# grouping by particular dataframe column
grouped = microbes.groupby(['Project Status'])
microbes_in_progress = grouped.get_group("In Progress")
microbes_in_progress.shape
```

[15]:
```
(1336, 19)
```

[16]:
```
# Split Dataframe using groupby() &
# grouping by particular dataframe column
grouped = microbes.groupby(['Project Status'])
microbes_complete = grouped.get_group("Complete")
microbes_complete.shape
```

[16]:
```
(1579, 19)
```

[17]:
```
#rename group of "complete" for ease
micro = microbes_complete
micro.shape
```

[17]:
```
(1579, 19)
```

[18]:
```
micro.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1579 entries, 0 to 2914
Data columns (total 19 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   HMP ID                   1579 non-null   int64
 1   GOLD ID                  1493 non-null   object
 2   Organism Name            1579 non-null   object
 3   Domain                   1552 non-null   object
 4   NCBI Superkingdom        1462 non-null   object
 5   HMP Isolation Body Site  1579 non-null   object
 6   Project Status           1579 non-null   object
 7   Current Finishing Level  1579 non-null   object
 8   NCBI Submission Status   1579 non-null   object
 9   NCBI Project ID          1579 non-null   int64
 10  Genbank ID               1579 non-null   object
```

```
 11  Gene Count               1579 non-null   int64
 12  IMG/HMP ID               1579 non-null   int64
 13  HOMD ID                  386 non-null    object
 14  Sequencing Center        1579 non-null   object
 15  Funding Source           1579 non-null   object
 16  Strain Repository ID     1272 non-null   object
 17  Unnamed: 17              0 non-null      float64
 18  Unnamed: 18              0 non-null      float64
dtypes: float64(2), int64(4), object(13)
memory usage: 246.7+ KB
```

I'm curious about some of the columns that have null values. The ones that are important to organism analysis are "Domain", and "NCBI Superkingdom".

[19]:
```
micro[['Domain','NCBI Superkingdom']].isnull().sum()
```

[19]:
```
Domain                27
NCBI Superkingdom    117
dtype: int64
```

[20]:
```
micro.groupby('Domain').count()
```

[20]:
```
            HMP ID  GOLD ID  Organism Name  NCBI Superkingdom
Domain
ARCHAEAL         2        2              2                  2  \
BACTERIAL     1541     1487           1541               1440
EUKARYAL         4        4              4                  4
VIRUS            5        0              5                  5

            HMP Isolation Body Site  Project Status  Current Finishing Level
Domain
ARCHAEAL                          2               2                        2  \
BACTERIAL                      1541            1541                     1541
EUKARYAL                          4               4                        4
VIRUS                             5               5                        5

            NCBI Submission Status  NCBI Project ID  Genbank ID  Gene Count
Domain
ARCHAEAL                         2                2           2           2  \
BACTERIAL                     1541             1541        1541        1541
EUKARYAL                         4                4           4           4
VIRUS                            5                5           5           5

            IMG/HMP ID  HOMD ID  Sequencing Center  Funding Source
Domain
ARCHAEAL             2        0                  2               2  \
BACTERIAL         1541      386               1541            1541
EUKARYAL             4        0                  4               4
```

```
VIRUS                5        0                   5               5

           Strain Repository ID  Unnamed: 17  Unnamed: 18
Domain
ARCHAEAL                      2            0            0
BACTERIAL                  1247            0            0
EUKARYAL                      2            0            0
VIRUS                         5            0            0
```

[21]:
```
micro.groupby('NCBI Superkingdom').count()
```

[21]:
```
                    HMP ID  GOLD ID  Organism Name  Domain
NCBI Superkingdom
Archaea                  2        2              2       2  \
Bacteria              1448     1384           1448    1437
Error!!!                 3        3              3       3
Eukaryota                4        4              4       4
Viruses                  5        0              5       5

                    HMP Isolation Body Site  Project Status
NCBI Superkingdom
Archaea                                   2               2  \
Bacteria                               1448            1448
Error!!!                                  3               3
Eukaryota                                 4               4
Viruses                                   5               5

                    Current Finishing Level  NCBI Submission Status
NCBI Superkingdom
Archaea                                   2                      2  \
Bacteria                               1448                   1448
Error!!!                                  3                      3
Eukaryota                                 4                      4
Viruses                                   5                      5

                    NCBI Project ID  Genbank ID  Gene Count  IMG/HMP ID
NCBI Superkingdom
Archaea                           2           2           2           2  \
Bacteria                       1448        1448        1448        1448
Error!!!                          3           3           3           3
Eukaryota                         4           4           4           4
Viruses                           5           5           5           5

                    HOMD ID  Sequencing Center  Funding Source
NCBI Superkingdom
Archaea                   0                  2               2  \
Bacteria                374               1448            1448
```

```
Error!!!                3                 3              3
Eukaryota               0                 4              4
Viruses                 0                 5              5

                   Strain Repository ID  Unnamed: 17  Unnamed: 18
NCBI Superkingdom
Archaea                               2            0            0
Bacteria                           1192            0            0
Error!!!                              3            0            0
Eukaryota                             2            0            0
Viruses                               5            0            0
```

There is an "Error!!!" value for Superkingdom, so that's nice to be able to see exactly what I should replace. I'll start by checking those values specfically.

[22]:
```
micro[micro['NCBI Superkingdom']=='Error!!!']
```

[22]:
```
      HMP ID  GOLD ID                       Organism Name     Domain
2478    9176  Gi05045          Streptococcus downei F0415  BACTERIAL  \
2481    9180  Gi05049  Streptococcus peroris ATCC 700780  BACTERIAL
2487    9192  Gi05061   Streptococcus vestibularis F0396  BACTERIAL

     NCBI Superkingdom HMP Isolation Body Site Project Status
2478          Error!!!                   oral       Complete  \
2481          Error!!!                   oral       Complete
2487          Error!!!                   oral       Complete

          Current Finishing Level
2478  Level 2: High-Quality Draft  \
2481  Level 2: High-Quality Draft
2487  Level 2: High-Quality Draft

                             NCBI Submission Status  NCBI Project ID
2478  6. annotation (and sequence) public on NCBI site           53567  \
2481  6. annotation (and sequence) public on NCBI site           53059
2487  6. annotation (and sequence) public on NCBI site           53573

        Genbank ID  Gene Count  IMG/HMP ID        HOMD ID
2478  AEKN00000000        2204   649990005  HOMD: tax_594  \
2481  AEVF00000000        1638   649990011  HOMD: tax_728
2487  AEKO00000000        1979   649990017   HOMD: tax_21

               Sequencing Center               Funding Source
2478   J. Craig Venter Institute  NIH-HMP Jumpstart Supplement  \
2481  Baylor College of Medicine                      NIH-HMP
2487   J. Craig Venter Institute                   NIH-NIAID
```

```
     Strain Repository ID  Unnamed: 17  Unnamed: 18
2478           BEI HM-475          NaN          NaN
2481          ATCC 700780          NaN          NaN
2487           BEI HM-561          NaN          NaN
```

All three are in the Bacterial domain, so I can replace their values with "Bacteria"

[23]:
```
micro['NCBI Superkingdom'].replace('Error!!!', 'Bacteria', inplace=True)
```

I can infer the Domain based on the Superkingdom and vice versa, but I can't use any values where both are missing, so I'll check those.

[24]:
```
len(micro.loc[micro['Domain'].isnull()& micro['NCBI Superkingdom'].isnull()])
```

[24]:
```
16
```

There are 16 values that have both missing so I'm going to drop those.

[25]:
```
micro=micro.drop(micro[(micro['Domain'].isnull()) & (micro['NCBI Superkingdom'].
  ↪isnull())].index)
micro.shape
```

[25]:
```
(1563, 19)
```

In order to replace the other values, I'm going to transform them to NaN first.

[26]:
```
micro['NCBI Superkingdom'].fillna('NaN', inplace=True)
```

[27]:
```
(micro['NCBI Superkingdom'] == "NaN").value_counts()
```

[27]:
```
NCBI Superkingdom
False    1462
True      101
Name: count, dtype: int64
```

[28]:
```
micro['Domain'].fillna('NaN', inplace=True)
```

[29]:
```
#check value counts
(micro['Domain'] == "NaN").value_counts()
```

[29]:
```
Domain
False    1552
True       11
Name: count, dtype: int64
```

I'm going to replace all of the Domain values with their relative Superkingdom name where applicable using pandas transform function.

[30]:
```
#make dataframe containing only rows with NaN in Domain or Superkingdom
micro_null = micro[(micro['Domain'] == "NaN") | (micro['NCBI Superkingdom'] ==␣
 ↪"NaN")]
```

[31]:
```
#See which rows have NaN to compare with their Superkingdom value
micro_null.loc[micro_null['Domain'] == "NaN"]
```

[31]:

```
      HMP ID GOLD ID                                Organism Name Domain  \
1314    1978     NaN             Actinomyces graevenitzii F0530    NaN
1463    2128     NaN                Arthrobacter albus DNF00011    NaN
1464    2129     NaN       Corynebacterium tuscaniense DNF00037    NaN
1465    2130     NaN               Oligella urethralis DNF00040    NaN
1467    2132     NaN  Prevotella histicola JCM 15637 = DNF00424    NaN
1469    2134     NaN          Peptoniphilus lacrimalis DNF00528    NaN
1470    2135     NaN       Staphylococcus haemolyticus DNF00585    NaN
1471    2136     NaN                  Prevotella bivia DNF00650    NaN
1472    2137     NaN               Prevotella buccalis DNF00853    NaN
1474    2139     NaN              Prevotella denticola DNF00960    NaN
1475    2140     NaN               Prevotella buccalis DNF00985    NaN

     NCBI Superkingdom HMP Isolation Body Site Project Status  \
1314          Bacteria                    oral       Complete
1463          Bacteria        urogenital_tract       Complete
1464          Bacteria        urogenital_tract       Complete
1465          Bacteria        urogenital_tract       Complete
1467          Bacteria        urogenital_tract       Complete
1469          Bacteria        urogenital_tract       Complete
1470          Bacteria        urogenital_tract       Complete
1471          Bacteria        urogenital_tract       Complete
1472          Bacteria        urogenital_tract       Complete
1474          Bacteria        urogenital_tract       Complete
1475          Bacteria        urogenital_tract       Complete

          Current Finishing Level  \
1314  Level 2: High-Quality Draft
1463  Level 2: High-Quality Draft
1464  Level 2: High-Quality Draft
1465  Level 2: High-Quality Draft
1467  Level 2: High-Quality Draft
1469  Level 2: High-Quality Draft
1470  Level 2: High-Quality Draft
1471  Level 2: High-Quality Draft
1472  Level 2: High-Quality Draft
1474  Level 2: High-Quality Draft
1475  Level 2: High-Quality Draft

                                NCBI Submission Status  NCBI Project ID
```

```
1314  6. annotation (and sequence) public on NCBI site          198881  \
1463  6. annotation (and sequence) public on NCBI site          219659
1464  6. annotation (and sequence) public on NCBI site          219660
1465  6. annotation (and sequence) public on NCBI site          219661
1467  6. annotation (and sequence) public on NCBI site          219666
1469  6. annotation (and sequence) public on NCBI site          219668
1470  6. annotation (and sequence) public on NCBI site          219669
1471  6. annotation (and sequence) public on NCBI site          219670
1472  6. annotation (and sequence) public on NCBI site          219672
1474  6. annotation (and sequence) public on NCBI site          219676
1475  6. annotation (and sequence) public on NCBI site          219677

        Genbank ID  Gene Count  IMG/HMP ID HOMD ID
1314  AWSC01000000        1897           0     NaN  \
1463  JRNH00000000        1596           0     NaN
1464  JRNG00000000        1792           0     NaN
1465  JRNI00000000        2118           0     NaN
1467  JRNJ00000000        2358           0     NaN
1469  JRNL00000000        3237           0     NaN
1470  JRNK00000000        2173           0     NaN
1471  JRNM00000000        1973           0     NaN
1472  JRNN00000000        2232           0     NaN
1474  JRNO00000000        2355           0     NaN
1475  JRNP00000000        2062           0     NaN

                                 Sequencing Center
1314  Washington University Genome Sequencing Center  \
1463                       J. Craig Venter Institute
1464                       J. Craig Venter Institute
1465                       J. Craig Venter Institute
1467                       J. Craig Venter Institute
1469                       J. Craig Venter Institute
1470                       J. Craig Venter Institute
1471                       J. Craig Venter Institute
1472                       J. Craig Venter Institute
1474                       J. Craig Venter Institute
1475                       J. Craig Venter Institute

                Funding Source Strain Repository ID  Unnamed: 17  Unnamed: 18
1314  NIH-HMP Sequencing Center          BEI HM-1132          NaN          NaN
1463                    NIH-HMP          BEI HM-1152          NaN          NaN
1464                    NIH-HMP          BEI HM-1153          NaN          NaN
1465                    NIH-HMP          BEI HM-1154          NaN          NaN
1467                    NIH-HMP       BEI Processing          NaN          NaN
1469                    NIH-HMP          BEI HM-1161          NaN          NaN
1470                    NIH-HMP          BEI HM-1164          NaN          NaN
1471                    NIH-HMP          BEI HM-1165          NaN          NaN
```

```
1472                  NIH-HMP          BEI HM-1169          NaN          NaN
1474                  NIH-HMP          BEI HM-1173          NaN          NaN
1475                  NIH-HMP          BEI HM-1174          NaN          NaN
```

[32]:
```
#count nulls to compare
(micro['Domain'] == "NaN").sum()
```

[32]:
```
11
```

[33]:
```
#Replace NaN values with "BACTERIAL"
micro["Domain"] = micro['Domain'].replace(["NaN"], "BACTERIAL")
micro
```

[33]:
```
      HMP ID  GOLD ID                                Organism Name     Domain  \
0          1  Gi03551          Abiotrophia defectiva ATCC 49176  BACTERIAL
1          4  Gi03555       Achromobacter piechaudii ATCC 43553  BACTERIAL
2          5  Gi03554            Achromobacter xylosoxidans C54  BACTERIAL
3         10  Gi03422        Acinetobacter baumannii ATCC 19606  BACTERIAL
4         12  Gi03421       Acinetobacter calcoaceticus RUH2202  BACTERIAL
…       …       …                                        …        …
2910    9995  Gi08654       Staphylococcus epidermidis NIHLM095  BACTERIAL
2911    9996  Gi09593  Aggregatibacter actinomycetemcomitans Y4  BACTERIAL
2912    9997  Gi09594               Corynebacterium durum F0235  BACTERIAL
2913    9998  Gi09595    Peptostreptococcus anaerobius VPI 4330  BACTERIAL
2914    9999  Gi09596  Prevotella sp. oral taxon 473 str. F0040  BACTERIAL

     NCBI Superkingdom HMP Isolation Body Site Project Status
0             Bacteria                    oral       Complete  \
1             Bacteria                airways       Complete
2             Bacteria                airways       Complete
3             Bacteria        urogenital_tract       Complete
4             Bacteria                    skin       Complete
…                …                       …             …
2910          Bacteria                unknown       Complete
2911          Bacteria                   oral       Complete
2912          Bacteria                   oral       Complete
2913          Bacteria                   oral       Complete
2914          Bacteria                   oral       Complete

                  Current Finishing Level
0     Level 3: Improved-High-Quality Draft  \
1             Level 2: High-Quality Draft
2        Level 5: Non-contiguous Finished
3             Level 2: High-Quality Draft
4             Level 2: High-Quality Draft
…                                       …
2910          Level 2: High-Quality Draft
```

```
2911           Level 2: High-Quality Draft
2912           Level 2: High-Quality Draft
2913           Level 2: High-Quality Draft
2914           Level 2: High-Quality Draft

                                NCBI Submission Status  NCBI Project ID
0     6. annotation (and sequence) public on NCBI site            33011  \
1     6. annotation (and sequence) public on NCBI site            46343
2     6. annotation (and sequence) public on NCBI site            38739
3     6. annotation (and sequence) public on NCBI site            38509
4     6. annotation (and sequence) public on NCBI site            38337
…                                                   …              …
2910  6. annotation (and sequence) public on NCBI site            62345
2911  6. annotation (and sequence) public on NCBI site            67199
2912  6. annotation (and sequence) public on NCBI site            67201
2913  6. annotation (and sequence) public on NCBI site            67203
2914  6. annotation (and sequence) public on NCBI site            67205

        Genbank ID  Gene Count  IMG/HMP ID        HOMD ID
0     ACIN00000000        1950   643886181  HOMD: tax_389  \
1     ADMS00000000        5755   647000200            NaN
2     ACRC00000000        6010           0  HOMD: tax_343
3     ACQB00000000        3832   647533101  HOMD: tax_554
4     ACPK00000000        3632   646206267            NaN
…              …           …           …              …
2910  AKGI00000000        2300           0            NaN
2911  AMEN00000000        2343           0            NaN
2912  AMEM00000000        2823           0            NaN
2913  AMEL00000000        1933           0            NaN
2914  AMEK00000000        2317           0            NaN

                                    Sequencing Center
0     Washington University Genome Sequencing Center  \
1                         Baylor College of Medicine
2                                   Broad Institute
3                                   Broad Institute
4                                   Broad Institute
…                                                 …
2910         NIH Intramural Sequencing Center (NISC)
2911  Washington University Genome Sequencing Center
2912  Washington University Genome Sequencing Center
2913  Washington University Genome Sequencing Center
2914  Washington University Genome Sequencing Center

                    Funding Source             Strain Repository ID
0     NIH-HMP Jumpstart Supplement           ATCC 49176, CIP 103242  \
1     NIH-HMP Jumpstart Supplement  ATCC 43553, CIP 55774, LMG 6100
```

```
2     NIH-HMP Jumpstart Supplement                    BEI HM-235
3     NIH-HMP Jumpstart Supplement          ATCC 19606, DSM 6974
4     NIH-HMP Jumpstart Supplement                    LMG 10517
…                             …                            …
2910         NIH-HMP Demo Projects                    BEI HM-909
2911     NIH-HMP Sequencing Center                    ATCC 43718
2912     NIH-HMP Sequencing Center                    BEI HM-755
2913     NIH-HMP Sequencing Center                    ATCC 27337
2914     NIH-HMP Sequencing Center                    BEI HM-756

      Unnamed: 17  Unnamed: 18
0             NaN          NaN
1             NaN          NaN
2             NaN          NaN
3             NaN          NaN
4             NaN          NaN
…             …            …
2910          NaN          NaN
2911          NaN          NaN
2912          NaN          NaN
2913          NaN          NaN
2914          NaN          NaN

[1563 rows x 19 columns]
```

[34]:
```
#Check that values replaced
(micro['Domain'] == "NaN").sum()
```

[34]:
```
0
```

Then I'll replace all of the Superkingdom values with their Domain name where applicable using pandas transform function.

[35]:
```
#Check values for Domain in regard to Superkingdom NaNs
kingdom = micro_null.loc[micro_null['NCBI Superkingdom'] == "NaN"]
kingdom['Domain'].value_counts()
```

[35]:
```
Domain
BACTERIAL    101
Name: count, dtype: int64
```

All of the missing Superkingdom values are in the Bacterial domain, so we can replace them with the relative value of "Bacteria".

[36]:
```
#count nulls to compare
(micro_null['NCBI Superkingdom'] == "NaN").sum()
```

[36]:
```
101
```

[37]:
```
#Replace NaN values with "BACTERIAL"
micro["NCBI Superkingdom"] = micro['NCBI Superkingdom'].replace(['NaN'],␣
 ↪'Bacteria')
micro
```

[37]:

```
      HMP ID  GOLD ID                                Organism Name     Domain  \
0          1  Gi03551          Abiotrophia defectiva ATCC 49176  BACTERIAL
1          4  Gi03555       Achromobacter piechaudii ATCC 43553  BACTERIAL
2          5  Gi03554            Achromobacter xylosoxidans C54  BACTERIAL
3         10  Gi03422        Acinetobacter baumannii ATCC 19606  BACTERIAL
4         12  Gi03421       Acinetobacter calcoaceticus RUH2202  BACTERIAL
…       …       …                                        …        …
2910    9995  Gi08654       Staphylococcus epidermidis NIHLM095  BACTERIAL
2911    9996  Gi09593  Aggregatibacter actinomycetemcomitans Y4  BACTERIAL
2912    9997  Gi09594               Corynebacterium durum F0235  BACTERIAL
2913    9998  Gi09595    Peptostreptococcus anaerobius VPI 4330  BACTERIAL
2914    9999  Gi09596  Prevotella sp. oral taxon 473 str. F0040  BACTERIAL

     NCBI Superkingdom HMP Isolation Body Site Project Status
0             Bacteria                   oral       Complete  \
1             Bacteria                 airways       Complete
2             Bacteria                 airways       Complete
3             Bacteria        urogenital_tract       Complete
4             Bacteria                   skin       Complete
…                  …                      …             …
2910          Bacteria                unknown       Complete
2911          Bacteria                   oral       Complete
2912          Bacteria                   oral       Complete
2913          Bacteria                   oral       Complete
2914          Bacteria                   oral       Complete

                  Current Finishing Level
0     Level 3: Improved-High-Quality Draft  \
1              Level 2: High-Quality Draft
2         Level 5: Non-contiguous Finished
3              Level 2: High-Quality Draft
4              Level 2: High-Quality Draft
…                                      …
2910           Level 2: High-Quality Draft
2911           Level 2: High-Quality Draft
2912           Level 2: High-Quality Draft
2913           Level 2: High-Quality Draft
2914           Level 2: High-Quality Draft

                              NCBI Submission Status  NCBI Project ID
0     6. annotation (and sequence) public on NCBI site            33011  \
1     6. annotation (and sequence) public on NCBI site            46343
```

```
2     6. annotation (and sequence) public on NCBI site            38739
3     6. annotation (and sequence) public on NCBI site            38509
4     6. annotation (and sequence) public on NCBI site            38337
…                                                  …              …
2910  6. annotation (and sequence) public on NCBI site            62345
2911  6. annotation (and sequence) public on NCBI site            67199
2912  6. annotation (and sequence) public on NCBI site            67201
2913  6. annotation (and sequence) public on NCBI site            67203
2914  6. annotation (and sequence) public on NCBI site            67205

        Genbank ID  Gene Count  IMG/HMP ID        HOMD ID
0     ACIN00000000        1950   643886181  HOMD: tax_389  \
1     ADMS00000000        5755   647000200            NaN
2     ACRC00000000        6010           0  HOMD: tax_343
3     ACQB00000000        3832   647533101  HOMD: tax_554
4     ACPK00000000        3632   646206267            NaN
…            …         …          …              …
2910  AKGI00000000        2300           0            NaN
2911  AMEN00000000        2343           0            NaN
2912  AMEM00000000        2823           0            NaN
2913  AMEL00000000        1933           0            NaN
2914  AMEK00000000        2317           0            NaN

                                  Sequencing Center
0     Washington University Genome Sequencing Center  \
1                        Baylor College of Medicine
2                                  Broad Institute
3                                  Broad Institute
4                                  Broad Institute
…                                              …
2910         NIH Intramural Sequencing Center (NISC)
2911  Washington University Genome Sequencing Center
2912  Washington University Genome Sequencing Center
2913  Washington University Genome Sequencing Center
2914  Washington University Genome Sequencing Center

                    Funding Source             Strain Repository ID
0     NIH-HMP Jumpstart Supplement           ATCC 49176, CIP 103242  \
1     NIH-HMP Jumpstart Supplement  ATCC 43553, CIP 55774, LMG 6100
2     NIH-HMP Jumpstart Supplement                        BEI HM-235
3     NIH-HMP Jumpstart Supplement             ATCC 19606, DSM 6974
4     NIH-HMP Jumpstart Supplement                         LMG 10517
…                             …                               …
2910         NIH-HMP Demo Projects                        BEI HM-909
2911     NIH-HMP Sequencing Center                        ATCC 43718
2912     NIH-HMP Sequencing Center                        BEI HM-755
2913     NIH-HMP Sequencing Center                        ATCC 27337
```

```
2914     NIH-HMP Sequencing Center                          BEI HM-756

      Unnamed: 17  Unnamed: 18
0             NaN          NaN
1             NaN          NaN
2             NaN          NaN
3             NaN          NaN
4             NaN          NaN
…             …            …
2910          NaN          NaN
2911          NaN          NaN
2912          NaN          NaN
2913          NaN          NaN
2914          NaN          NaN

[1563 rows x 19 columns]
```

[38]:
```
#Check that values replaced
(micro['NCBI Superkingdom'] == "NaN").sum()
```

[38]:
```
0
```

0.0.3 Exploration

Then, do a graphical analysis creating a minimum of four graphs. Label your graphs appropriately and explain/analyze the information provided by each graph. Your analysis should begin to answer the question(s) you are addressing. Write a short overview/conclusion of the insights gained from your graphical analysis.

Now that those are cleaned up, I'm going to review the full dataset based on Gene Count to start.

[39]:
```
micro['Gene Count'].describe()
```

[39]:
```
count    1563.000000
mean     2729.550864
std      1288.903478
min         0.000000
25%      1956.000000
50%      2411.000000
75%      3176.000000
max      8490.000000
Name: Gene Count, dtype: float64
```

There are no non-null values for "Gene Count", but msome are counted as 0.

[40]:
```
micro_gene_count=micro[micro['Gene Count']==0]
micro_gene_count['NCBI Superkingdom'].value_counts()
```

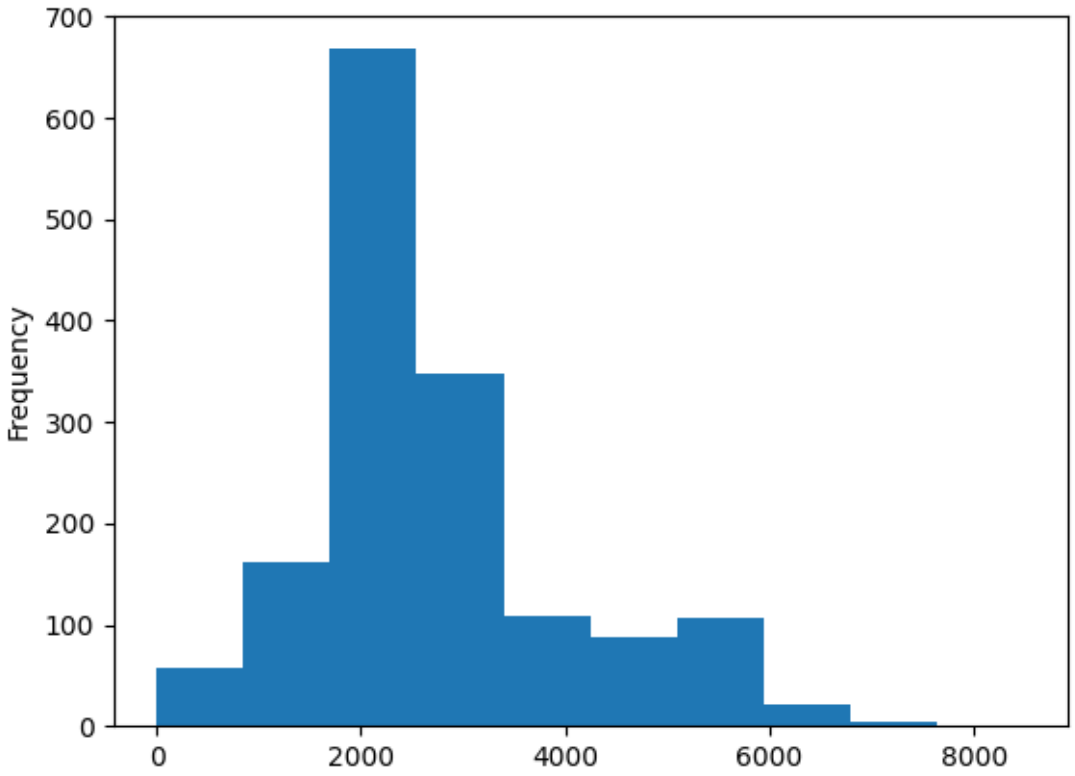
[40]:
```
NCBI Superkingdom
Bacteria     47
Viruses       5
Eukaryota     4
Name: count, dtype: int64
```

There are 47 bacteria, 5 viruses, and 4 eukaryota absent from the count. Because these may be based on a reporting error, I may want to drop these later to improve the model, but I'll keep them for now.

There are many species listed in this project, so I want to look at their distribution of gene count frequency.

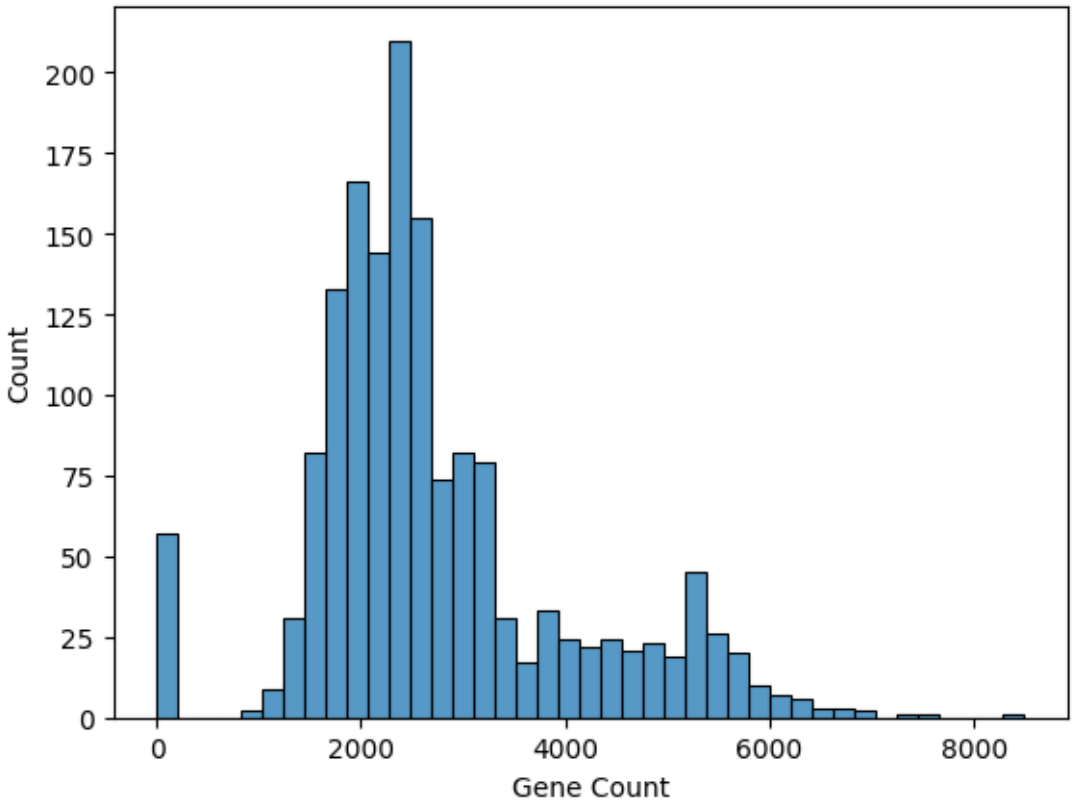
[41]:
```
micro["Gene Count"].plot(kind='hist')
```

[41]:
```
<Axes: ylabel='Frequency'>
```



[42]:
```
sns.histplot(data=micro, x="Gene Count")
```

[42]:
```
<Axes: xlabel='Gene Count', ylabel='Count'>
```



Interestingly, there is an almost normal distribution, skewed right, but we can see that the species with gene counts in the middle range have the highest freqency.

I'm curious about the microbe with the highest gene count (max value from the descriptive statistics), with a value of 8490.

[43]:
```
micro[micro['Gene Count']==8490]
```

[43]:
```
     HMP ID  GOLD ID             Organism Name     Domain NCBI Superkingdom
679    1211  Gi10716  Streptomyces sp. HGB0020  BACTERIAL          Bacteria  \

    HMP Isolation Body Site Project Status      Current Finishing Level
679  gastrointestinal_tract       Complete  Level 2: High-Quality Draft  \

                              NCBI Submission Status  NCBI Project ID
679  6. annotation (and sequence) public on NCBI site            72491  \

       Genbank ID  Gene Count  IMG/HMP ID HOMD ID Sequencing Center
679  AGER00000000        8490           0     NaN   Broad Institute  \

                Funding Source Strain Repository ID  Unnamed: 17  Unnamed: 18
679  NIH-HMP Sequencing Center           BEI HM-789          NaN          NaN
```

I want to check to see if there is another Streptomyces species with high prevalence.

[44]:
```
micro[micro['Organism Name'].str.contains("Streptomyces")]
```

[44]:
```
     HMP ID  GOLD ID             Organism Name     Domain NCBI Superkingdom
679    1211  Gi10716  Streptomyces sp. HGB0020  BACTERIAL          Bacteria  \
934    1486  Gi16997  Streptomyces sp. HPH0547  BACTERIAL          Bacteria

    HMP Isolation Body Site Project Status      Current Finishing Level
679  gastrointestinal_tract       Complete  Level 2: High-Quality Draft  \
934  gastrointestinal_tract       Complete  Level 2: High-Quality Draft

                             NCBI Submission Status  NCBI Project ID
679  6. annotation (and sequence) public on NCBI site            72491  \
934  6. annotation (and sequence) public on NCBI site           169487

       Genbank ID  Gene Count  IMG/HMP ID HOMD ID Sequencing Center
679  AGER00000000        8490           0     NaN   Broad Institute  \
934  ATCE00000000        6635           0     NaN   Broad Institute

                Funding Source Strain Repository ID  Unnamed: 17  Unnamed: 18
679  NIH-HMP Sequencing Center           BEI HM-789          NaN          NaN
934  NIH-HMP Sequencing Center           BEI HM-859          NaN          NaN
```

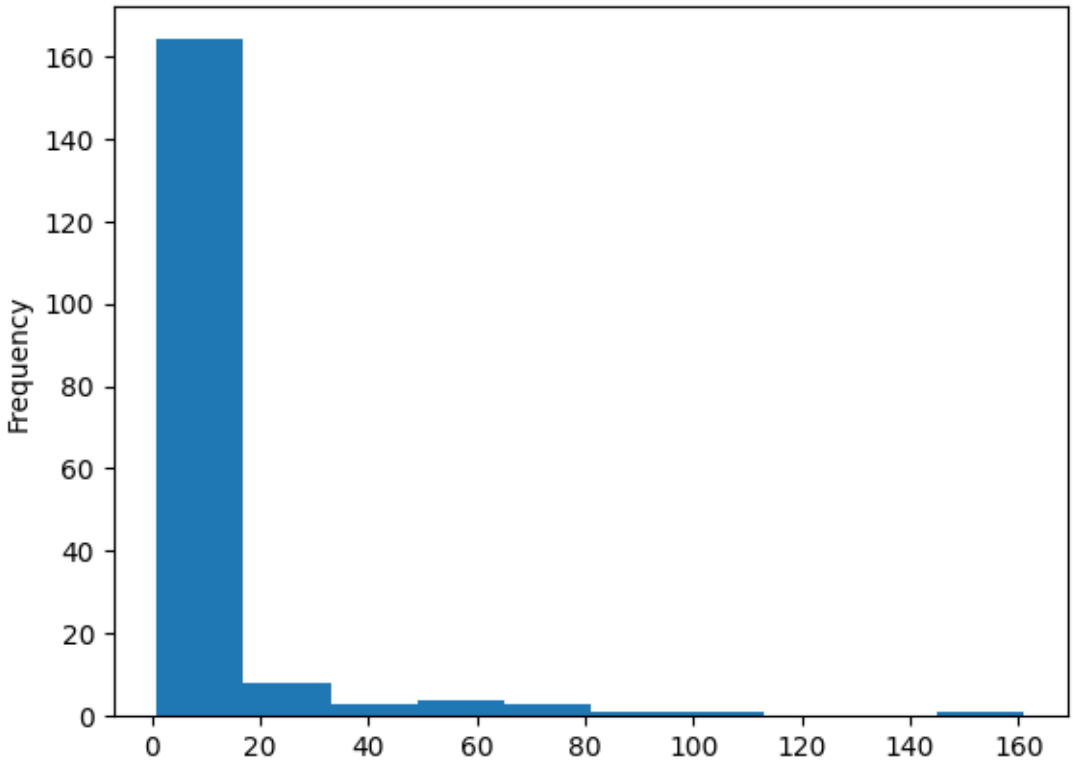
The presences of another Streoptomyces with a high gene count makes me think that it may be beneficial to sort organisms on their genus. I'm going make a new dataframe and attempt to add a column for genus by extracting the first word of the Organism Name.

[45]:
```
df = micro
df['Genus'] = df['Organism Name'].str.split(' ').str[0]
df['Genus'].nunique()
```

[45]:
```
185
```

[46]:
```
df["Genus"].value_counts().plot(kind='hist')
```

[46]:
```
<Axes: ylabel='Frequency'>
```



[47]:
```
#Seaborn plot - not as beneficial this time so I'm not using it
#plot frequency of genus for all entires
#x = df["Genus"].value_counts()
#sns.histplot(data=df, x=x)
```

Since there are so many distributed around 0-15, I'm going to exclude those and print the value counts of the higher ones.

[48]:
```
count = df[df.Genus.isin(df["Genus"].value_counts(dropna=False).loc[lambda x :␣
↪x>=16].index)].copy()
count["Genus"].value_counts()
```

[48]:
```
Genus
Streptococcus         161
Enterococcus          110
Propionibacterium      92
Lactobacillus          73
Helicobacter           70
Prevotella             65
Staphylococcus         64
Bacteroides            63
Escherichia            61
```

```
Clostridium           58
Corynebacterium       38
Fusobacterium         36
Actinomyces           34
Bifidobacterium       31
Treponema             25
Gardnerella           22
Klebsiella            21
Eubacterium           21
Neisseria             19
Porphyromonas         17
Capnocytophaga        17
Veillonella           16
Name: count, dtype: int64
```

Since the list is limited, I could have guessed the index number until I got to the value I wanted.

[49]:
```
df['Genus'].value_counts(ascending=False)[:22]
```

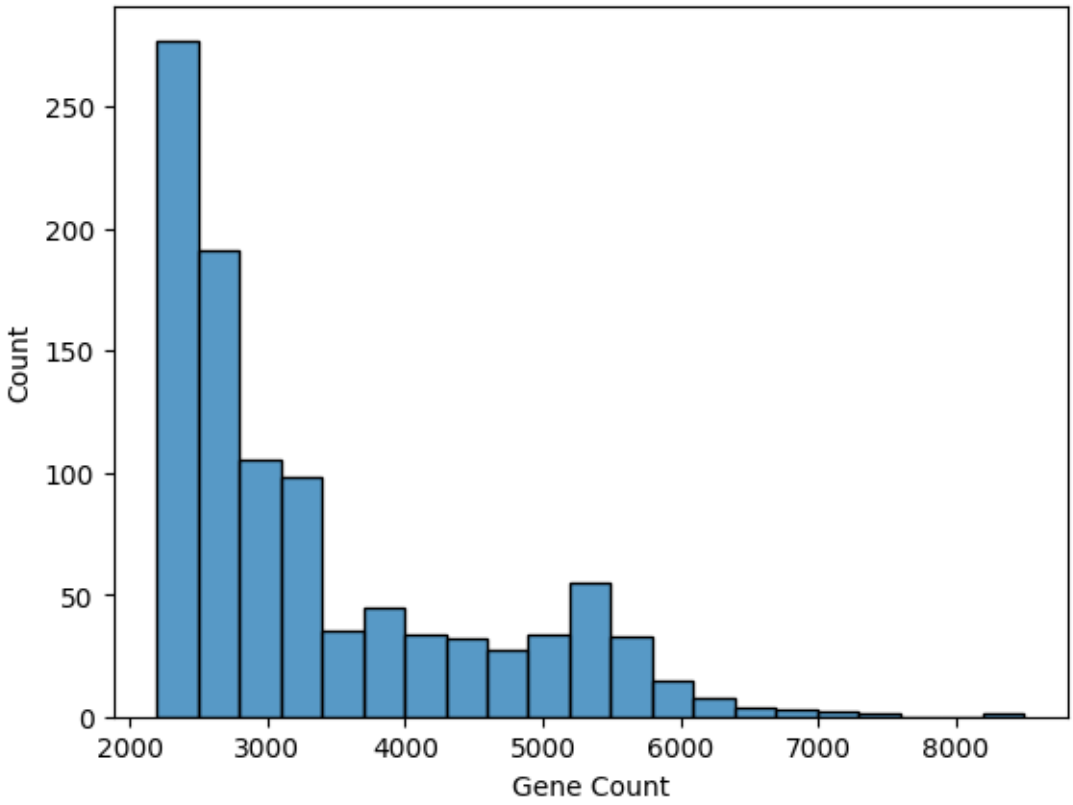
[49]:
```
Genus
Streptococcus        161
Enterococcus         110
Propionibacterium     92
Lactobacillus         73
Helicobacter          70
Prevotella            65
Staphylococcus        64
Bacteroides           63
Escherichia           61
Clostridium           58
Corynebacterium       38
Fusobacterium         36
Actinomyces           34
Bifidobacterium       31
Treponema             25
Gardnerella           22
Klebsiella            21
Eubacterium           21
Neisseria             19
Capnocytophaga        17
Porphyromonas         17
Veillonella           16
Name: count, dtype: int64
```

Let's see if this changes based on a subset of the most prevalent organisms.

[50]:
```
top_organisms=micro.sort_values(by='Gene Count', ascending = False)[:1000]
```

[51]:
```
sns.histplot(data=top_organisms, x="Gene Count")
```

[51]:
```
<Axes: xlabel='Gene Count', ylabel='Count'>
```



[52]:
```
# add genus column to the top_organisms
top = top_organisms
top['Genus'] = top['Organism Name'].str.split(' ').str[0]
```

[53]:
```
top['Genus'].value_counts()[:22]
```

[53]:
```
Genus
Enterococcus          109
Propionibacterium      91
Staphylococcus         62
Bacteroides            62
Escherichia            60
Clostridium            57
Streptococcus          53
Prevotella             51
Corynebacterium        32
Treponema              24
```

```
Lactobacillus         22
Klebsiella            21
Fusobacterium         20
Actinomyces           17
Neisseria             17
Capnocytophaga        16
Parabacteroides       15
Acinetobacter         14
Bifidobacterium       12
Providencia           11
Eubacterium            9
Selenomonas            8
Name: count, dtype: int64
```

In the original histogram, we saw that the highest frequency of species was between 1800-2400 gene count, so I am going to make a dataframe around that.

[54]:
```
mid_microbe = micro[(micro['Gene Count'].values >= 1800) & (micro['Gene Count'].
↪values <= 2400)]
```

[55]:
```
mid_microbe['Genus'] = mid_microbe['Organism Name'].str.split(' ').str[0]
```

[56]:
```
mid_microbe["Genus"].value_counts().sort_values(ascending=False)[:22]
```

[56]:
```
Genus
Streptococcus            142
Lactobacillus             30
Staphylococcus            27
Prevotella                25
Corynebacterium           23
Bifidobacterium           21
Propionibacterium         20
Fusobacterium             19
Actinomyces               14
Veillonella               13
Porphyromonas             10
Selenomonas               10
Haemophilus               10
Mobiluncus                 8
Capnocytophaga             5
Anaerococcus               5
Neisseria                  4
Peptostreptococcaceae      4
Helicobacter               4
Oribacterium               4
Leptotrichia               4
Peptoniphilus              4
```

```
Name: count, dtype: int64
```

Now that I'm comfortable with having added "Genus" to my dataframes, I'm going to replace the main dataframe with the amended one.

[57]:
```
df = micro
```

I want to know how many unique sites on the human body were researched.

[58]:
```
micro['HMP Isolation Body Site'].nunique()
```

[58]:
```
12
```

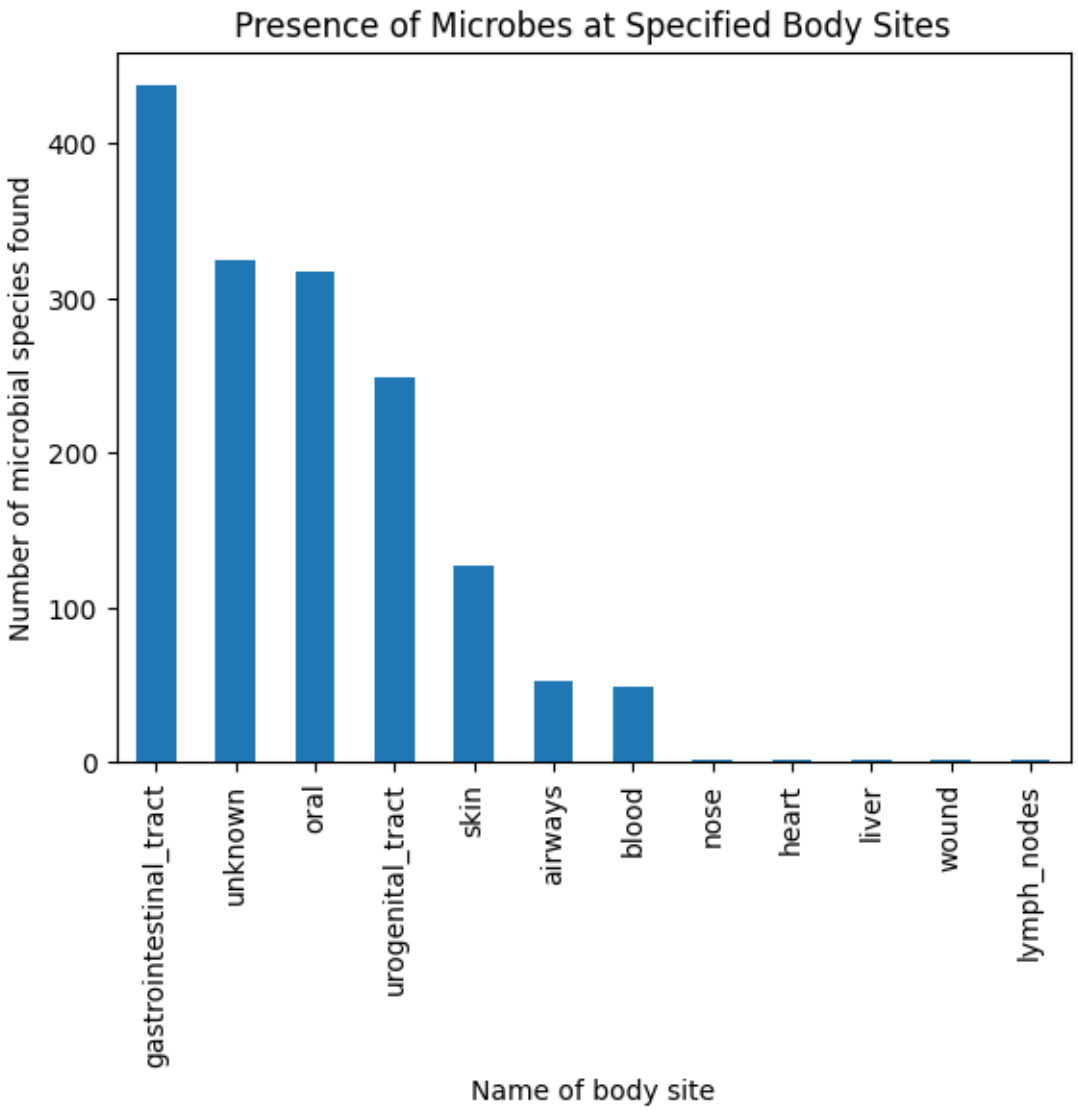
[59]:
```
micro['HMP Isolation Body Site'].value_counts()
```

[59]:
```
HMP Isolation Body Site
gastrointestinal_tract    437
unknown                   324
oral                      317
urogenital_tract          249
skin                      127
airways                    53
blood                      49
nose                        2
heart                       2
liver                       1
wound                       1
lymph_nodes                 1
Name: count, dtype: int64
```

Here is a chart of the species diversity at the different sites.

[60]:
```
micro['HMP Isolation Body Site'].value_counts().plot(kind='bar')
plt.title('Presence of Microbes at Specified Body Sites')
plt.ylabel('Number of microbial species found')
plt.xlabel('Name of body site')
```

[60]:
```
Text(0.5, 0, 'Name of body site')
```



To find out more about the kingdom variance throughout the body, I'll look into those values.

[61]:
```
micro.groupby('NCBI Superkingdom')['HMP Isolation Body Site'].nunique().
  ↪sort_values(ascending=False)
```

[61]:
```
NCBI Superkingdom
Bacteria     11
Eukaryota     3
Archaea       1
Viruses       1
Name: HMP Isolation Body Site, dtype: int64
```

The Bacteria Kingdom is most prevelent throughout the body, so I'm going to look more closely

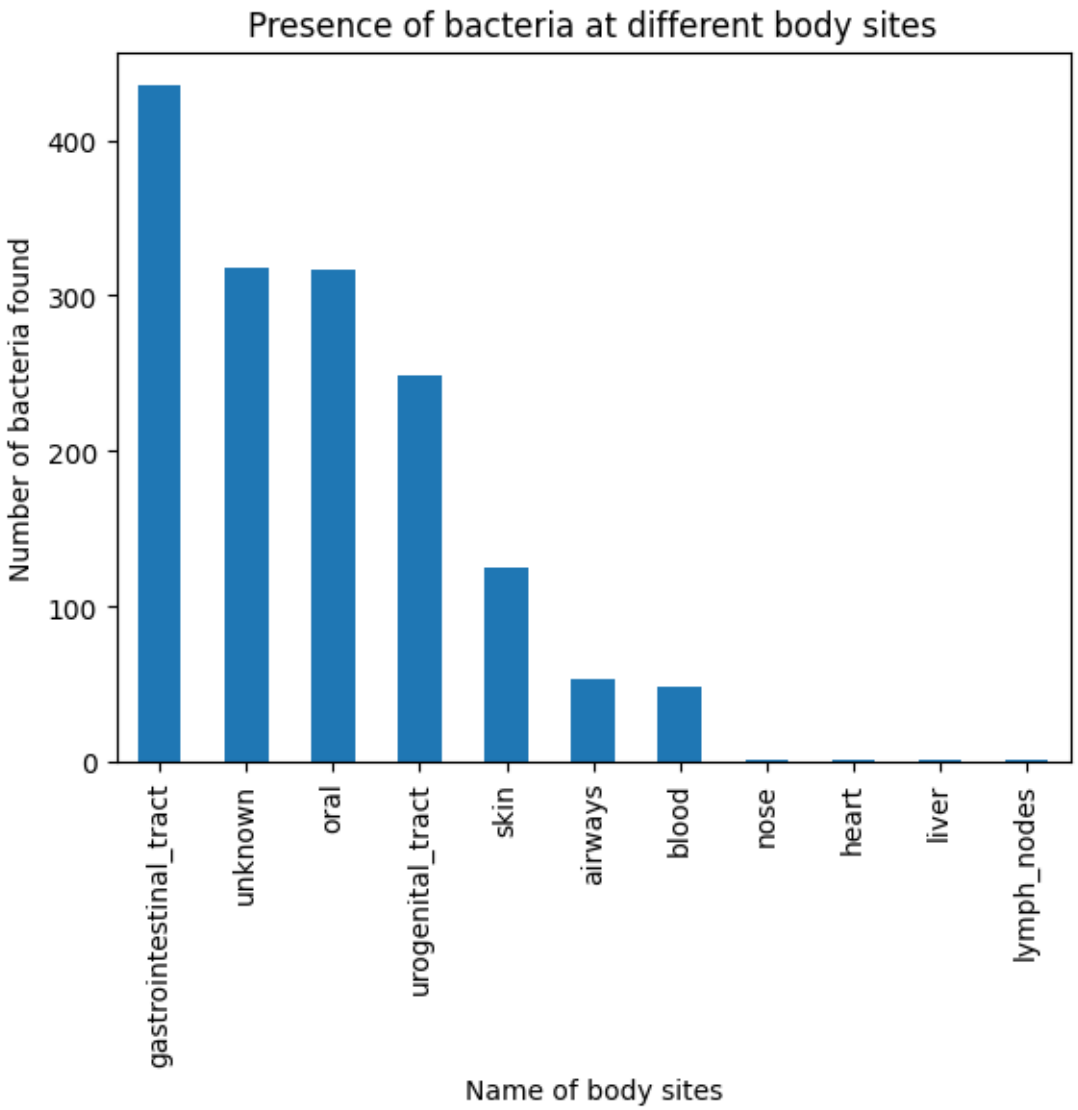
that their locations by making a dataframe with the values from the “bacterial” domain.

[62]:
```
#Select Bacterial domain and check body sites
bac=micro.loc[micro['Domain']=='BACTERIAL']
bac['HMP Isolation Body Site'].unique()
```

[62]:
```
array(['oral', 'airways', 'urogenital_tract', 'skin',
       'gastrointestinal_tract', 'blood', 'unknown', 'liver', 'nose',
       'heart', 'lymph_nodes'], dtype=object)
```

[63]:
```
#Plot
bac['HMP Isolation Body Site'].value_counts(ascending=False).plot(kind='bar')
plt.ylabel('Number of bacteria found')
plt.xlabel('Name of body sites')
plt.title('Presence of bacteria at different body sites')
```

[63]:
```
Text(0.5, 1.0, 'Presence of bacteria at different body sites')
```



The greatest amount of bacteria are found in the gastrointestinal tract. Let's look at the other kingdoms.

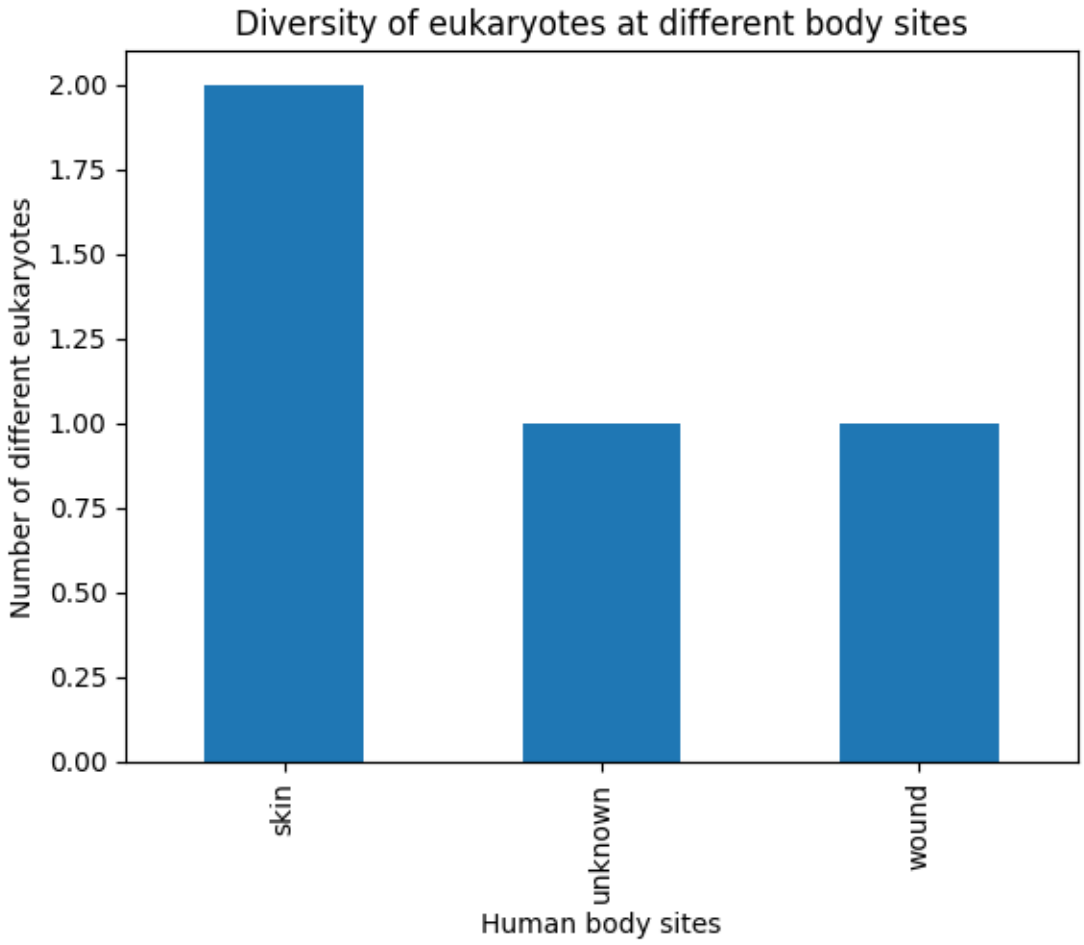
[64]:
```
#Select Eukaryal domain and check body sites
euk=micro.loc[micro['Domain']=='EUKARYAL']
euk['HMP Isolation Body Site'].unique()
```

[64]:
```
array(['unknown', 'skin', 'wound'], dtype=object)
```

[65]:
```
#plot
euk['HMP Isolation Body Site'].value_counts(ascending=False).plot(kind='bar')
plt.ylabel('Number of different eukaryotes')
plt.xlabel('Human body sites')
```

```
plt.title('Diversity of eukaryotes at different body sites')
```

[65]:
```
Text(0.5, 1.0, 'Diversity of eukaryotes at different body sites')
```



The greatest amount of eukaryotes are found on the skin.

[66]:
```
vir=micro.loc[micro['Domain']=='VIRUS']
vir['HMP Isolation Body Site'].unique()
```

[66]:
```
array(['unknown'], dtype=object)
```

From the data that we have, we are unable to determine where the greatest number of viruses are located. I'm guessing this is because viruses infect and replicate in cells, sometimes mainly infecting neighboring cells, but often spreading throughout the body.

[67]:
```
arc=micro.loc[micro['Domain']=='ARCHAEAL']
arc['HMP Isolation Body Site'].unique()
```

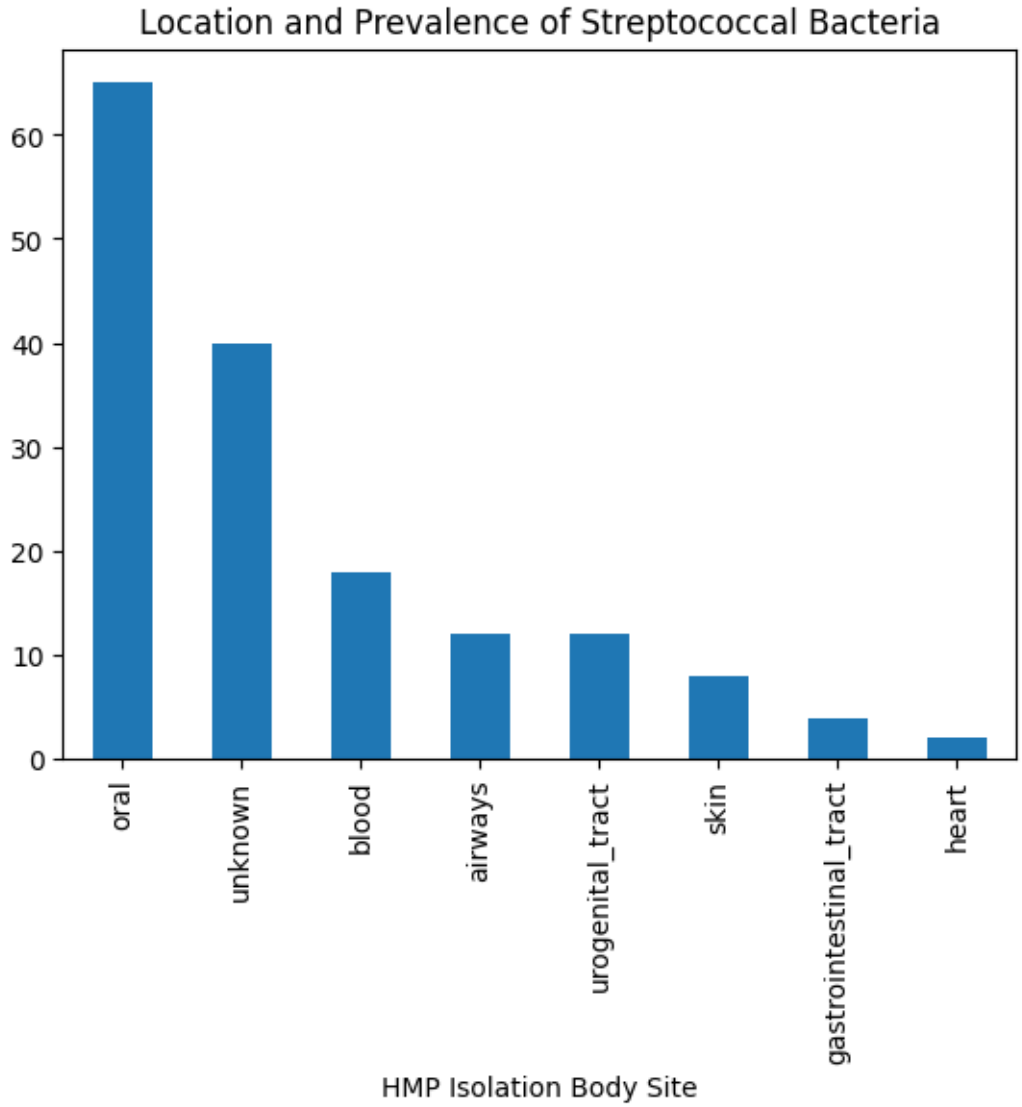
[67]:
```
array(['gastrointestinal_tract'], dtype=object)
```

Based on this project, archaea are found solely in the Gastrointestinal Tract.

Streptococcus are the most prevalent throughout the body. Let's look further into this.

[68]:
```
strep=micro.loc[micro['Genus']=='Streptococcus']
strep['HMP Isolation Body Site'].value_counts().plot(kind='bar')
plt.title("Location and Prevalence of Streptococcal Bacteria")
```

[68]:
```
Text(0.5, 1.0, 'Location and Prevalence of Streptococcal Bacteria')
```

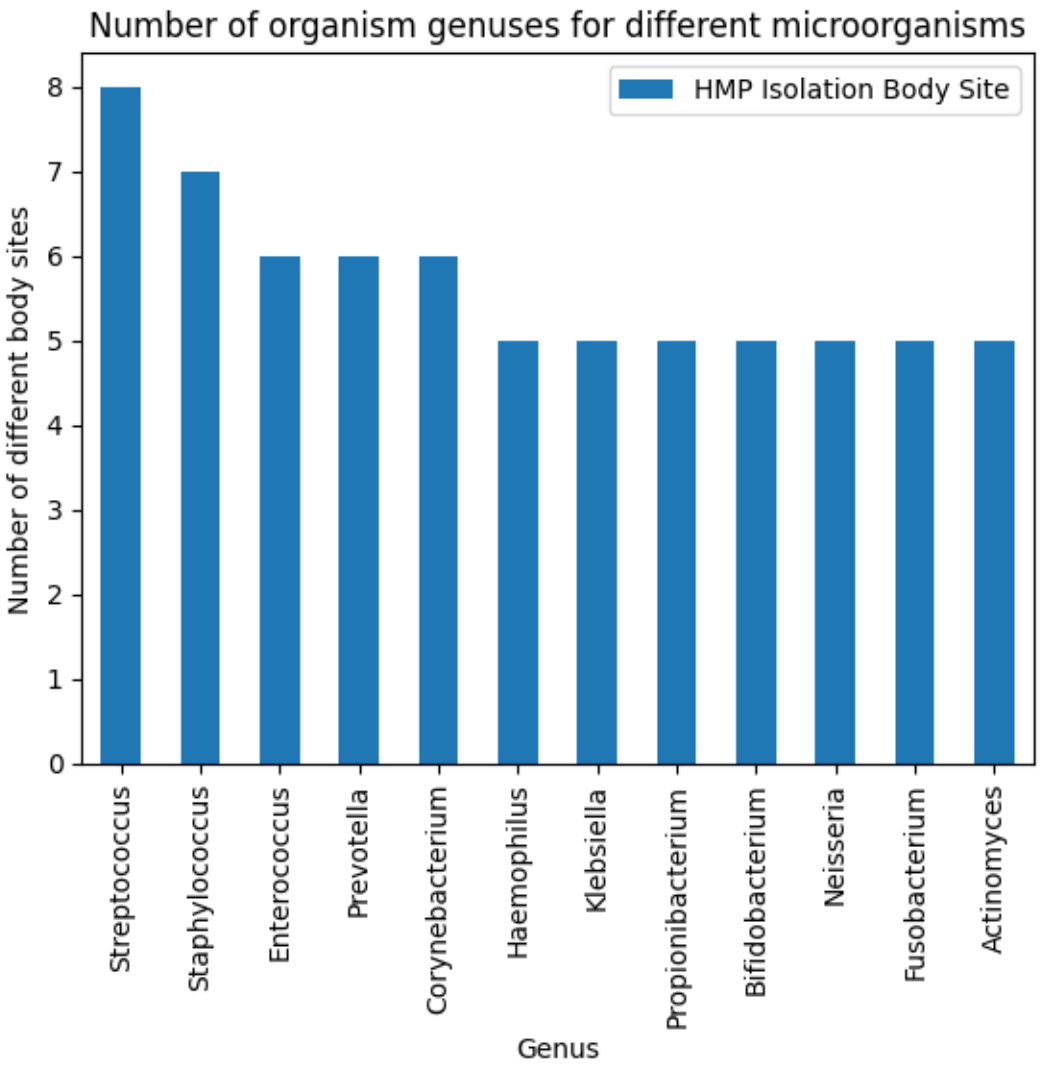


Let's see how other top genuses compare.

[69]:
```
genus_site=micro.groupby('Genus')['HMP Isolation Body Site'].nunique().
  ↪sort_values(ascending=False)
genus_df=pd.DataFrame(genus_site)
top_genus=genus_df[genus_df['HMP Isolation Body Site']>4]
print(top_genus)
top_genus.plot(kind='bar')
plt.ylabel('Number of different body sites')
plt.title('Number of organism genuses for different microorganisms')
```

```
                  HMP Isolation Body Site
Genus
Streptococcus                           8
Staphylococcus                          7
Enterococcus                            6
Prevotella                              6
Corynebacterium                         6
Haemophilus                             5
Klebsiella                              5
Propionibacterium                       5
Bifidobacterium                         5
Neisseria                               5
Fusobacterium                           5
Actinomyces                             5
```

[69]: Text(0.5, 1.0, 'Number of organism genuses for different microorganisms')



Because viruses are limited, we can print all of their names.

[70]:
```
viruses= micro[micro['NCBI Superkingdom'] =='Viruses']
viruses['Organism Name']
```

[70]:
```
2852    Pseudomonas phage F_HA0480sp/Pa1651
2853                Pseudomonas phage JBD18
2854                Pseudomonas phage JBD25
2855                Pseudomonas phage JBD26
2856                Pseudomonas phage JBD67
Name: Organism Name, dtype: object
```

Because eukaryotes are limited, we can print all of their names.

[71]:
```
eukaryotes= micro[micro['NCBI Superkingdom']=='Eukaryota']
eukaryotes['Organism Name']
```

[71]:
```
601                  Exophiala dermatitidis NIH/UT8656
983                    Phialophora europaea CBS 101466
985     Mucor circinelloides f. circinelloides 1006PhL
1065                   Sporothrix schenckii ATCC 58251
Name: Organism Name, dtype: object
```

Because archaea are limited, we can print all of their names.

[72]:
```
archaea= micro[micro['NCBI Superkingdom']=='Archaea']
archaea['Organism Name']
```

[72]:
```
302    Methanobrevibacter smithii DSM 2374
303    Methanobrevibacter smithii DSM 2375
Name: Organism Name, dtype: object
```

[73]:
```
micro['NCBI Superkingdom'].value_counts()
```

[73]:
```
NCBI Superkingdom
Bacteria     1552
Viruses         5
Eukaryota       4
Archaea         2
Name: count, dtype: int64
```

Observations from this EDA: - Gastrointestinal system shows most diversity of microbes - Streptomyces sp. HGB0020 shows the maximum gene count in human - Streptococcus is most common genus

Milestone 2 *I made a lot of transformations previously because I felt like they were important for visualizations, but there are still more I can do for preparing a model. My goal is to use a predictive model to figure out which organisms are likely to be present in various locations of the body (outlined in the data as "HMP Isolation Body Sites") and, in theory, that would be used to make predictions on how to restore the microbiome of a specific patient, based on the sequencing of their sample.*

[74]:
```
# change df name for transformations
df = micro
```

[75]:
```
df.head()
```

[75]:
```
   HMP ID  GOLD ID                          Organism Name     Domain
0       1  Gi03551      Abiotrophia defectiva ATCC 49176  BACTERIAL  \
1       4  Gi03555  Achromobacter piechaudii ATCC 43553  BACTERIAL
2       5  Gi03554        Achromobacter xylosoxidans C54  BACTERIAL
3      10  Gi03422   Acinetobacter baumannii ATCC 19606  BACTERIAL
```

```
4      12  Gi03421  Acinetobacter calcoaceticus RUH2202  BACTERIAL

  NCBI Superkingdom HMP Isolation Body Site Project Status
0          Bacteria                   oral       Complete  \
1          Bacteria                airways       Complete
2          Bacteria                airways       Complete
3          Bacteria        urogenital_tract       Complete
4          Bacteria                   skin       Complete

               Current Finishing Level
0  Level 3: Improved-High-Quality Draft  \
1           Level 2: High-Quality Draft
2      Level 5: Non-contiguous Finished
3           Level 2: High-Quality Draft
4           Level 2: High-Quality Draft

                           NCBI Submission Status  NCBI Project ID
0  6. annotation (and sequence) public on NCBI site            33011  \
1  6. annotation (and sequence) public on NCBI site            46343
2  6. annotation (and sequence) public on NCBI site            38739
3  6. annotation (and sequence) public on NCBI site            38509
4  6. annotation (and sequence) public on NCBI site            38337

     Genbank ID  Gene Count  IMG/HMP ID        HOMD ID
0  ACIN00000000        1950   643886181  HOMD: tax_389  \
1  ADMS00000000        5755   647000200            NaN
2  ACRC00000000        6010           0  HOMD: tax_343
3  ACQB00000000        3832   647533101  HOMD: tax_554
4  ACPK00000000        3632   646206267            NaN

                              Sequencing Center
0  Washington University Genome Sequencing Center  \
1                     Baylor College of Medicine
2                                Broad Institute
3                                Broad Institute
4                                Broad Institute

                Funding Source             Strain Repository ID  Unnamed: 17
0  NIH-HMP Jumpstart Supplement           ATCC 49176, CIP 103242          NaN  \
1  NIH-HMP Jumpstart Supplement  ATCC 43553, CIP 55774, LMG 6100          NaN
2  NIH-HMP Jumpstart Supplement                      BEI HM-235          NaN
3  NIH-HMP Jumpstart Supplement             ATCC 19606, DSM 6974          NaN
4  NIH-HMP Jumpstart Supplement                         LMG 10517          NaN

   Unnamed: 18          Genus
0          NaN    Abiotrophia
1          NaN  Achromobacter
```

```
2          NaN  Achromobacter
3          NaN  Acinetobacter
4          NaN  Acinetobacter
```

[76]:
```
#find na values
df.isna().sum()
```

[76]:
```
HMP ID                        0
GOLD ID                      70
Organism Name                 0
Domain                        0
NCBI Superkingdom             0
HMP Isolation Body Site       0
Project Status                0
Current Finishing Level       0
NCBI Submission Status        0
NCBI Project ID               0
Genbank ID                    0
Gene Count                    0
IMG/HMP ID                    0
HOMD ID                    1177
Sequencing Center             0
Funding Source                0
Strain Repository ID        296
Unnamed: 17                1563
Unnamed: 18                1563
Genus                         0
dtype: int64
```

[77]:
```
#check percentage missing values
round((df.isnull().sum() * 100/ len(df)),2).sort_values(ascending=False)
```

[77]:
```
Unnamed: 18                100.00
Unnamed: 17                100.00
HOMD ID                     75.30
Strain Repository ID        18.94
GOLD ID                      4.48
HMP ID                       0.00
Funding Source               0.00
Sequencing Center            0.00
IMG/HMP ID                   0.00
Gene Count                   0.00
Genbank ID                   0.00
NCBI Project ID              0.00
NCBI Submission Status       0.00
Current Finishing Level      0.00
Project Status               0.00
```

```
HMP Isolation Body Site      0.00
NCBI Superkingdom            0.00
Domain                       0.00
Organism Name                0.00
Genus                        0.00
dtype: float64
```

[78]:
```
# HOMD ID has 75% missing values so I'm checking it
df['HOMD ID']
```

[78]:
```
0       HOMD: tax_389
1                 NaN
2       HOMD: tax_343
3       HOMD: tax_554
4                 NaN
            …
2910              NaN
2911              NaN
2912              NaN
2913              NaN
2914              NaN
Name: HOMD ID, Length: 1563, dtype: object
```

The unnamed columns provide no information so they can be removed. HOMD ID isn't necessary for analysis so it can be removed as well.

[79]:
```
df2 = df.drop(['Unnamed: 18', 'Unnamed: 17', 'HOMD ID'], axis=1)
```

[80]:
```
#check percentage missing values
round((df2.isnull().sum() * 100/ len(df2)),2).sort_values(ascending=False)
```

[80]:
```
Strain Repository ID      18.94
GOLD ID                    4.48
HMP ID                     0.00
NCBI Project ID            0.00
Funding Source             0.00
Sequencing Center          0.00
IMG/HMP ID                 0.00
Gene Count                 0.00
Genbank ID                 0.00
NCBI Submission Status     0.00
Current Finishing Level    0.00
Project Status             0.00
HMP Isolation Body Site    0.00
NCBI Superkingdom          0.00
Domain                     0.00
Organism Name              0.00
```

```
Genus                          0.00
dtype: float64
```

[81]:
```
df['Strain Repository ID']
```

[81]:
```
0                ATCC 49176, CIP 103242
1       ATCC 43553, CIP 55774, LMG 6100
2                             BEI HM-235
3                  ATCC 19606, DSM 6974
4                              LMG 10517
                        …
2910                          BEI HM-909
2911                          ATCC 43718
2912                          BEI HM-755
2913                          ATCC 27337
2914                          BEI HM-756
Name: Strain Repository ID, Length: 1563, dtype: object
```

[82]:
```
df["GOLD ID"]
```

[82]:
```
0       Gi03551
1       Gi03555
2       Gi03554
3       Gi03422
4       Gi03421
          …
2910    Gi08654
2911    Gi09593
2912    Gi09594
2913    Gi09595
2914    Gi09596
Name: GOLD ID, Length: 1563, dtype: object
```

[83]:
```
#HOMD ID and Strain Repository ID are not necessary for analysis so they can be␣
 ↪dropped
df2 = df2.drop(['Strain Repository ID', 'GOLD ID'], axis=1)
```

[84]:
```
df2.columns.tolist()
```

[84]:
```
['HMP ID',
 'Organism Name',
 'Domain',
 'NCBI Superkingdom',
 'HMP Isolation Body Site',
 'Project Status',
 'Current Finishing Level',
 'NCBI Submission Status',
```

```
 'NCBI Project ID',
 'Genbank ID',
 'Gene Count',
 'IMG/HMP ID',
 'Sequencing Center',
 'Funding Source',
 'Genus']
```

[85]:
```
#removing the rest of the ID columns
df2 = df2.drop(['NCBI Project ID',
 'Genbank ID', 'IMG/HMP ID'], axis=1)
```

[86]:
```
#new df name to preserve previous
df = df2
```

[87]:
```
df.columns.tolist()
```

[87]:
```
['HMP ID',
 'Organism Name',
 'Domain',
 'NCBI Superkingdom',
 'HMP Isolation Body Site',
 'Project Status',
 'Current Finishing Level',
 'NCBI Submission Status',
 'Gene Count',
 'Sequencing Center',
 'Funding Source',
 'Genus']
```

[88]:
```
# subset of df
test_df = df[['HMP ID',
 'Organism Name',
 'Domain',
 'NCBI Superkingdom',
 'HMP Isolation Body Site',
 'Gene Count',
 'Genus']]
```

[89]:
```
test_df.shape
```

[89]:
```
(1563, 7)
```

[90]:
```
#checking size and unique values of columns
df['Genus'].value_counts()
```

[90]:
```
Genus
Streptococcus        161
Enterococcus         110
Propionibacterium     92
Lactobacillus         73
Helicobacter          70
                     …
Pediococcus            1
Mycobacterium          1
Micrococcus            1
Leuconostoc            1
Acetobacteraceae       1
Name: count, Length: 185, dtype: int64
```

[91]:
```
# rename for to preserve previous
new_df = test_df
```

[92]:
```
new_df
```

[92]:
```
      HMP ID                              Organism Name     Domain  \
0          1          Abiotrophia defectiva ATCC 49176  BACTERIAL
1          4       Achromobacter piechaudii ATCC 43553  BACTERIAL
2          5            Achromobacter xylosoxidans C54  BACTERIAL
3         10        Acinetobacter baumannii ATCC 19606  BACTERIAL
4         12       Acinetobacter calcoaceticus RUH2202  BACTERIAL
…      …                                          …         …
2910    9995       Staphylococcus epidermidis NIHLM095  BACTERIAL
2911    9996  Aggregatibacter actinomycetemcomitans Y4  BACTERIAL
2912    9997               Corynebacterium durum F0235  BACTERIAL
2913    9998    Peptostreptococcus anaerobius VPI 4330  BACTERIAL
2914    9999  Prevotella sp. oral taxon 473 str. F0040  BACTERIAL

     NCBI Superkingdom HMP Isolation Body Site  Gene Count              Genus
0             Bacteria                   oral        1950         Abiotrophia
1             Bacteria                airways        5755       Achromobacter
2             Bacteria                airways        6010       Achromobacter
3             Bacteria       urogenital_tract        3832       Acinetobacter
4             Bacteria                   skin        3632       Acinetobacter
…                  …                      …           …                  …
2910          Bacteria                unknown        2300      Staphylococcus
2911          Bacteria                   oral        2343     Aggregatibacter
2912          Bacteria                   oral        2823     Corynebacterium
2913          Bacteria                   oral        1933  Peptostreptococcus
2914          Bacteria                   oral        2317          Prevotella

[1563 rows x 7 columns]
```

I want to remove Domain or Superkingdom because they seem to have the same information, but first I need to check that. I'm going to convert the strings to lowercase so I may make a clean comparison. Then I'll search for the root words in the other column and see if any are not the same (check if they are all duplicated).

[93]:
```
new_df['Domain'] = new_df['Domain'].str.lower()
new_df['Domain']
```

[93]:
```
0       bacterial
1       bacterial
2       bacterial
3       bacterial
4       bacterial
          …
2910    bacterial
2911    bacterial
2912    bacterial
2913    bacterial
2914    bacterial
Name: Domain, Length: 1563, dtype: object
```

[94]:
```
new_df['NCBI Superkingdom'] = new_df['NCBI Superkingdom'].str.lower()
new_df['NCBI Superkingdom']
```

[94]:
```
0       bacteria
1       bacteria
2       bacteria
3       bacteria
4       bacteria
          …
2910    bacteria
2911    bacteria
2912    bacteria
2913    bacteria
2914    bacteria
Name: NCBI Superkingdom, Length: 1563, dtype: object
```

[95]:
```
# use apply to find if the "superkingdom" string is in "domain", if it is not,␣
 ↪return NaN.
new_df['New'] = new_df.apply(lambda x: x['NCBI Superkingdom'] if x['NCBI␣
 ↪Superkingdom'] in
                     x['Domain'] else np.nan, axis=1)
```

[96]:
```
#check NA value total
new_df['New'].isna().sum()
```

[96]:
```
9
```

[97]:
```
#There are 9 null values so we can see review the entries manually
new_df[new_df['New'].isna()]
```

[97]:

	HMP ID	Organism Name	Domain	\
601	1120	Exophiala dermatitidis NIH/UT8656	eukaryal	\
983	1541	Phialophora europaea CBS 101466	eukaryal	
985	1544	Mucor circinelloides f. circinelloides 1006PhL	eukaryal	
1065	1624	Sporothrix schenckii ATCC 58251	eukaryal	
2852	9774	Pseudomonas phage F_HA0480sp/Pa1651	virus	
2853	9843	Pseudomonas phage JBD18	virus	
2854	9847	Pseudomonas phage JBD25	virus	
2855	9848	Pseudomonas phage JBD26	virus	
2856	9886	Pseudomonas phage JBD67	virus	

	NCBI Superkingdom	HMP Isolation Body Site	Gene Count	Genus	New
601	eukaryota	unknown	0	Exophiala	NaN
983	eukaryota	skin	0	Phialophora	NaN
985	eukaryota	skin	0	Mucor	NaN
1065	eukaryota	wound	0	Sporothrix	NaN
2852	viruses	unknown	0	Pseudomonas	NaN
2853	viruses	unknown	0	Pseudomonas	NaN
2854	viruses	unknown	0	Pseudomonas	NaN
2855	viruses	unknown	0	Pseudomonas	NaN
2856	viruses	unknown	0	Pseudomonas	NaN

I see that these are duplicates as well, so all of the values between "NCBI Superkingdom" and "Domain" are the same. I can remove one of them. I am choosing to drop "NCBI Superkingdom" as well as the "new" column I used for comparison purposes.

[98]:
```
new_df = new_df.drop(['NCBI Superkingdom', 'New'], axis=1)
```

[99]:
```
new_df.columns.to_list()
```

[99]:
```
['HMP ID',
 'Organism Name',
 'Domain',
 'HMP Isolation Body Site',
 'Gene Count',
 'Genus']
```

[100]:
```
new_df.shape
```

[100]:
```
(1563, 6)
```

[101]:
```
new_df["Organism Name"].nunique()
```

[101]:
```
1557
```

There are too many unique organisms to make them into dummies, so I'm going to try a couple of different approaches. I'll see how it looks to se Genus as predictor and HMP Body Site

[102]:
```
new_df["Genus"].nunique()
```

[102]:
```
185
```

#Make new df of dummy variables for all but the "HMP ID" (not helpful for modeling) and "Gene Count" (already numerical) set_df = pd.get_dummies(data=new_df, columns=['Domain', 'HMP Isolation Body Site', 'Genus', 'Organism Name'])

From milestone 2: These are the necessary columns for correlating and making predictions for regions of the body. I'm going to attempt to build a model by first labeling the predictor and outcome variables, then splitting the data and running a model. I was imagining a knearest neighbor model, but a logistic regression might be more appropriate… or I may be way off and in need of direction!

Milestone 3

0.1 KNN Classifier

[103]:
```
new_df.columns
```

[103]:
```
Index(['HMP ID', 'Organism Name', 'Domain', 'HMP Isolation Body Site',
       'Gene Count', 'Genus'],
      dtype='object')
```

0.1.1 Target is Genus

[104]:
```
#I selected this order of variables through trial & error based on the accuracy␣
 ↪below
genus_df = new_df[["Domain", "Gene Count", "HMP Isolation Body Site"]]
```

[105]:
```
#get dummy variables for model scaling/fitting
genus_df = pd.get_dummies(genus_df)
```

[107]:
```
X = genus_df #features
y = new_df["Genus"] #target
```

[108]:
```
#split data for training and testing
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=100)
```

[109]:
```
#Scale/fit/transform
scaler = StandardScaler()
scaler.fit(X_train)

X_train_scaled = scaler.transform(X_train) ### fit training data above
```

```
X_test_scaled = scaler.transform(X_test) ### ((fitting leaks information from␣
 ↪test into train))
```

The code below was an attempt to scale target values to see if that would be beneficial, but it is not.

[110]:
```
# y_train_scale = scaler.fit_transform(X_train) ### fit training data above <br>
# y_test_scale = scaler.transform(X_test) ### ((fitting leaks information from␣
 ↪test into train))
```

[111]:
```
#define classifier and fit
classifier = KNeighborsClassifier(n_neighbors=5)
classifier.fit(X_train_scaled, y_train)
```

[112]:
```
#calculate accuracy
y_pred = classifier.predict(X_test_scaled)
accuracy = accuracy_score(y_test, y_pred)
accuracy
```

I attempted to use hyperparamter tuning (below) based on Exercise 9.2 and it gave me some insight, even though it called an error because the least populated class in y only has 1 member.

[113]:
```
#adjust to match other assignment
standardizer = StandardScaler()
X_train = standardizer.fit_transform(X_train_scaled)
X_test = standardizer.transform(X_test_scaled)

#define classifier and build pipeline
knn = KNeighborsClassifier()
pipe = Pipeline([('standardizer', standardizer), ('classifier', knn)])
```

[114]:
```
# fit pipeline
pipe.fit(X_train_scaled, y_train)
```

[115]:
```
#score the knn model on the testing data
pipe.score(X_test_scaled, y_test)
```

[116]:
```
# define search space
search_space = [{"classifier__n_neighbors": range(1,20,1)}]
```

[117]:
```
#gridsearch has an error, but the process still works.
gridsearch = GridSearchCV(pipe, search_space, cv=5).fit(X_train_scaled, y_train)
gridsearch
```

[118]:
```
# get best parameters based
gridsearch.best_estimator_.get_params()["classifier"]
```

[119]:
```
# run pipeline with indicated number of neighbors
knn = KNeighborsClassifier(n_neighbors=9)
pipe = Pipeline([('standardizer', standardizer), ('classifier', knn)])
pipe.fit(X_train_scaled, y_train)
#score the knn model on the testing data
pipe.score(X_test_scaled,y_test)
```

The accuracy is higher. Still not great in any way, but good enough to move on.

0.1.2 Target is HMP Isolation Body Site

Repeat steps above with HMP Isolation Body Site as the target.

[120]:
```
body_df = new_df[["Genus","Domain", "Gene Count" ]]
```

[121]:
```
body_df = pd.get_dummies(body_df)
```

[122]:
```
X = body_df
y = new_df['HMP Isolation Body Site']
```

[123]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20,␣
↪random_state=100)
```

[124]:
```
scaler = StandardScaler()
scaler.fit(X_train)

X_train_scaled = scaler.transform(X_train) ### fit training data above
X_test_scaled = scaler.transform(X_test) ### ((fitting leaks information from␣
↪test into train))
```

[125]:
```
classifier = KNeighborsClassifier(n_neighbors=5)
classifier.fit(X_train_scaled, y_train)
```

[126]:
```
y_pred = classifier.predict(X_test_scaled)
```

[127]:
```
accuracy = accuracy_score(y_test, y_pred)
accuracy
```

I attempted to use hyperparamter tuning (below) based on Exercise 9.2 and it gave me some insight, even though it called an error because the least populated class in y only has 1 member.

[128]:
```
standardizer = scaler
X_train_scaled = standardizer.fit_transform(X_train)
X_test_scale = standardizer.transform(X_test)

knn = KNeighborsClassifier()
pipe = Pipeline([('standardizer', standardizer), ('classifier', knn)])
```

[129]:
```
pipe.fit(X_train_scaled, y_train)
```

[130]:
```
#score the knn model on the testing data
pipe.score(X_test_scaled,y_test)
```

[131]:
```
search_space = [{"classifier__n_neighbors": range(1,20,1)}]
```

[132]:
```
#gridsearch has an error, but the process still works.
gridsearch = GridSearchCV(pipe, search_space, cv=5).fit(X_train_scaled, y_train)
gridsearch
```

[133]:
```
gridsearch.best_estimator_.get_params()["classifier"]
```

[134]:
```
knn = KNeighborsClassifier(n_neighbors=10)
pipe = Pipeline([('standardizer', standardizer), ('classifier', knn)])
pipe.fit(X_train_scaled, y_train)
#score the knn model on the testing data
pipe.score(X_test_scaled,y_test)
```

This method of target variable is looking better!

0.2 Logistic Regression

Repeat the steps from KNN but use logistic regression with both types of features/targets.

0.2.1 Target is Genus

[135]:
```
genus_df = new_df[["Domain", "HMP Isolation Body Site", "Gene Count"]]
```

[136]:
```
genus_df = pd.get_dummies(genus_df)
```

[137]:
```
X = genus_df
y = new_df["Genus"]
```

[138]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
↪random_state=100)
```

[139]:
```
scaler = StandardScaler()
scaler.fit(X_train)

X_train_scaled = scaler.transform(X_train) ### fit training data above
X_test_scaled = scaler.transform(X_test) ### ((fitting leaks information from␣
↪test into train))
```

The code below was an attempt to scale target values to see if that would be beneficial, but it is not.

[140]:
```
#y_train_scale = scaler.fit_transform(X_train) ### fit training data above <br>
#y_test_scale = scaler.transform(X_test) ### ((fitting leaks information from␣
 ↪test into train))
```

[141]:
```
log_reg = LogisticRegression()
log_reg.fit(X_train_scaled, y_train)
```

[142]:
```
y_pred = log_reg.predict(X_test_scaled)
```

[143]:
```
accuracy = accuracy_score(y_test, y_pred)
accuracy
```

0.2.2 Target is body site

[144]:
```
body_df = new_df[["Genus","Domain", "Gene Count" ]]
```

[145]:
```
body_df = pd.get_dummies(body_df)
```

[146]:
```
X = body_df
y = new_df['HMP Isolation Body Site']
```

[147]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20,␣
 ↪random_state=100)
```

[148]:
```
scaler = StandardScaler()
scaler.fit(X_train)

X_train_scaled = scaler.transform(X_train) ### fit training data above
X_test_scaled = scaler.transform(X_test) ### ((fitting leaks information from␣
 ↪test into train))
```

[149]:
```
log_reg = LogisticRegression()
log_reg.fit(X_train_scaled, y_train)
```

[150]:
```
y_pred = log_reg.predict(X_test_scaled)
```

[151]:
```
accuracy = accuracy_score(y_test, y_pred)
accuracy
```

This is lower than I had hoped, but not terrible. I'm curious if I had used a larger dataset, if there would have had higher accuracy. Maybe I can try that again later. I'm not sure what to do, I pored over these models and tried rearranging, refitting, scaling, not scaling, etc. so I'm happy with these basics that are left. Building a model to predict the location based on the sample of bacteria seems like it has the most liklihood of being a good model. Looking forward to honing these skills!

0.3 Summary of Project

This was a fascinating exploration in data mining. I am really grateful to have been able to merge my previous research focus (microbiome health) with these new skills. I am quick to recognize how novice I am in this field and that my approaches are elementary, but hypothesizing about what discoveries could be made using machine learning in this field has been very stimulating for me. I completed a meta-analysis years ago in effort to apply the framework from ecosystem restoration principles of ecology to restoration of the human gut. There was little, if not no, discussion about this approach to healing and health within the medical field. A sentiment analysis may have been a good method for that work, but as I've become more interested in the biological aspect, I chose to look at the organisms themselves. Since the publication of that project years ago, the Human Microbiome Project has expanded their database exponentially. The data is collected from two primary sites and is obviously not comprehensive of the population as whole. That alone is a huge ethical implication for using this data to make predictions on the population as a whole. There were a few things to consider with the steps I took in my data process, one of them being the exclusion of all data that was not marked as a completed "Project Status." In my cleaning process, I felt like having these values removed would make this practice experience less prone to error. Given that the accuracy of my models was not as high as I would have liked, I don't know if this is true. Ideally, I would go back and include this data to run the models again, to see if the expanded training data would allow for better prediction of the test data. With this in consideration, I would say the model is not ready to be deployed, but the revisions for the dataset seems to have been appropriately applied, so there may be more efficacy if I expanded the number of points to include the incomplete set, or modeled it a bit differently with better understanding of the models' capabilities/other models that could be applied. I also would like to explore other methods of measuring accuracy as, pointed out by Professor Werner, there are many ways to evaluate a model. Overall this was a super fun experience. I really enjoyed the cleaning and transformation stages, and am happy that the exploration and modeling are becoming more enjoyable. I'm certainly gaining more proficiency and am looking forward to honing my skills further.

Thank you for this excellent course, Professor Werner!