Counting on The Norton Anthology of American Literature

December 12, 2023

This notebook shows calculations for the quantities cited in the essay "Counting on *The Norton Anthology of American Literature*." Other quantities cited in the article refer to external sources such as Open Syllabus. We only specify here figures derived from our database.

It begins with an alphabetical list of "the Norton 103," the authors who were selected for the first edition of the *The Norton Anthology of American Literature* (*NAAL*) and have been reselected for every subsequent edition. Then, the document proceeds by showing the calculations for each quantitative claim in the essay in the order that they appear.

0.1 Methodology

While we believe that ours is the best existing relational database representing *NAAL*'s tables of contents, we do not claim that it is a perfect representation. Read the essay for a thorough description of the choices made for the database.

We used Django to create and manage a PostgreSQL relational database. Then, we customized the Django admin to enable multiple contributors to add works, authors, and anthologies to the database simultaneously.

The database describes the anthologies' selections as Norton's editors present them in the tables of contents (TOC). We do not impose relationships of excerption not otherwise described by the editors. For example, while some poets' works are attributed to specific volumes of their poetry (e.g., Whitman's are attributed to sections of *Leaves of Grass*), others' poems are not (e.g., Stevens's are not attributed to *Harmonium*).

We work backward from the most contemporary TOC entries (as of this writing, the tenth edition of 2022), aggregating prior occurrences of a text and/or author to their most current bibliographic representations. For example, Sagoyewatha appears in the eighth edition under the name Red Jacket, but as Sagoyewatha in the current edition. We reconcile all works historically attributed to Red Jacket to Sagoyewatha to avoid creating the false impression that the same text has two distinct authors.

0.2 Data

We are actively working to expand the database. Eventually, we hope to produce a web interface for users to query, subset, and export data. Scholars interested in working with the data beforehand can email the authors, Erik Fredner (`fredner@virginia.edu`) and J.D. Porter (`porterjd@sas.upenn.edu`).

1 The Norton 103

The authors listed below were selected for the first edition of the *NAAL* in 1979 and reselected for every subsequent edition. Careful readers will note that there are in fact 105 names below. This is because Norton editors have represented Alexander Hamilton, John Jay, and James Madison as the collective author of *The Federalist Papers*. Because editors have attributed *The Federalist Papers* to all three, we treat them as a single author entity in our database.

> Henry Adams; A. R. Ammons; Sherwood Anderson; John Ashbery; James Baldwin; Imamu Amiri Baraka (LeRoi Jones); Saul Bellow; John Berryman; Ambrose Bierce; Elizabeth Bishop; William Bradford; Anne Bradstreet; Gwendolyn Brooks; William Cullen Bryant; William Byrd; Willa Cather; Charles W. Chesnutt; Kate Chopin; James Fenimore Cooper; Hart Crane; Stephen Crane; J. Hector St. John de Crèvecœur; Countee Cullen; E. E. Cummings; H. D. (Hilda Doolittle); Emily Dickinson; John Dos Passos; Frederick Douglass; Theodore Dreiser; W. E. B. Du Bois; Jonathan Edwards; T. S. Eliot; Ralph Ellison; Ralph Waldo Emerson; William Faulkner; F. Scott Fitzgerald; Benjamin Franklin; Mary E. Wilkins Freeman; Philip Freneau; Robert Frost; Margaret Fuller; Hamlin Garland; Allen Ginsberg; Alexander Hamilton, John Jay, and James Madison; Joel Chandler Harris; Bret Harte; Nathaniel Hawthorne; Ernest Hemingway; Langston Hughes; Washington Irving; Henry James; Randall Jarrell; Thomas Jefferson; Sarah Orne Jewett; Sarah Kemble Knight; Denise Levertov; Abraham Lincoln; Jack London; Henry Wadsworth Longfellow; Robert Lowell; Cotton Mather; Herman Melville; James Merrill; W. S. Merwin; Edna St. Vincent Millay; Marianne Moore; Flannery O'Connor; Frank O'Hara; Charles Olson; Thomas Paine; Sylvia Plath; Edgar Allan Poe; Katherine Anne Porter; Ezra Pound; Adrienne Rich; Edwin Arlington Robinson; Theodore Roethke; Philip Roth; Mary Rowlandson; Carl Sandburg; Anne Sexton; Gary Snyder; Gertrude Stein; John Steinbeck; Wallace Stevens; Harriet Beecher Stowe; Edward Taylor; Henry David Thoreau; Jean Toomer; Mark Twain (Samuel L. Clemens); John Updike; Booker T. Washington; Eudora Welty; Edith Wharton; Phillis Wheatley; Walt Whitman; John Greenleaf Whittier; Richard Wilbur; William Carlos Williams; John Winthrop; John Woolman; James Wright; and Richard Wright.

2 Calculations

All of the calculations below are preceded by a header containing the sentence(s) from the article in which they appear. The calculations that were used to generate the figures referenced in those sentence(s) follow. The calculations are presented in the order in which these statements appear in the essay.

2.1 Read data

[1]:
```
import pandas as pd

df = pd.read_csv("naal.csv") # naal.csv was exported from the database
cols = [
    "work_id",
    "work_title",
    "parent_id",
```

```
    "author_id",
    "anthology_id",
    "author_name",
    "gender_id",
    "race_id",
    "birth_year",
    "race_name",
    "gender_name",
    "anthology_title",
]
df.columns = cols
```

[2]:
```
# (naal edition #, database id #)
anth2id = [
    (1, 15),
    (2, 14),
    (3, 16),
    (4, 13),
    (5, 12),
    (6, 11),
    (7, 8),
    (8, 6),
    (9, 10),
    (10, 18),
]
```

[3]:
```
l = list()

for x in df["anthology_id"]:
    for y in anth2id:
        if x == y[1]:
            l.append(y[0])
```

[4]:
```
df["naal_edition"] = l
```

2.2 Of works cut from one edition to the next, a majority are always cut from authors who stay in the anthology. Only 20% of works cut from one edition to the next are by authors who are removed entirely.

[5]:
```
l = list()

for x in range(1, 10):
    dff = df[df["naal_edition"] == x]
    dfg = df[df["naal_edition"] == x + 1]
    dff_works = set(dff["work_id"]) - set(
        dff["parent_id"]
    )  # see below for explanation re: parent works
```

```
    dfg_works = set(dfg["work_id"]) - set(dfg["parent_id"])
    retained = set(dff["author_id"]) & set(dfg["author_id"])
    cut = set(dff["author_id"]) - set(dfg["author_id"])
    works_cut = dff_works - dfg_works
    cut_df = dff[dff["work_id"].isin(works_cut)]
    # output
    d = dict()
    d["naal edition 1"] = x
    d["naal edition 2"] = x + 1
    d["works cut"] = len(works_cut)
    d["works cut from retained authors"] = (
        cut_df[cut_df["author_id"].isin(retained)]["work_id"].unique().size
    )
    d["works cut from cut authors"] = (
        cut_df[cut_df["author_id"].isin(cut)]["work_id"].unique().size
    )
    l.append(d)
```

[6]:
```
cuts = pd.DataFrame(l)
cuts["% cut from retained authors"] = cuts["works cut from retained authors"].
 ↪divide(
    cuts["works cut"]
)
cuts["% cut from cut authors"] = cuts["works cut from cut authors"].divide(
    cuts["works cut"]
)
```

[7]:
```
%%capture cap --no-stderr
print(cuts.to_markdown(index=False))
```

[8]:
```
from IPython.display import display, Markdown
display(Markdown(cap.stdout))
```

naal edition 1	naal edition 2	works cut	works cut from retained authors	works cut from cut authors	% cut from retained authors	% cut from cut authors
1	2	351	327	24	0.931624	0.0683761
2	3	150	132	18	0.88	0.12
3	4	148	123	25	0.831081	0.168919
4	5	388	353	35	0.909794	0.0902062
5	6	144	125	19	0.868056	0.131944
6	7	308	225	83	0.730519	0.269481
7	8	175	122	53	0.697143	0.302857
8	9	241	145	96	0.60166	0.39834
9	10	276	196	80	0.710145	0.289855

[9]:
```
cuts["works cut from cut authors"].sum() / cuts["works cut"].sum()
```

[9]: 0.19853278312700595

[10]:
```
cuts["works cut from retained authors"].sum() / cuts["works cut"].sum()
```

[10]: 0.801467216872994

2.3 Table 1

Table 1 shows the number of authors and works selected per edition. Note that "works" are defined as all works within a given edition that are *not* parent works within that edition. This is because *NAAL* TOCs frequently create representations like the following:

Uncle Tom's Cabin; or, Life among the Lowly

Volume I

Chapter I. In Which the Reader Is Introduced to a Man of Humanity

Neither *Uncle Tom's Cabin* nor Volume I contain any text in the anthology except for what is in its excerpts (of which Chapter I is one example).

We nevertheless describe parent works in the database in order to establish relationships of excerption such that we can say that some part of *Uncle Tom's Cabin* has appeared in every edition, though the specific selections have changed.

When counting works in an edition, we count only those that do not appear as parents of another work within that same edition.

[11]:
```
l = list()

for x in range(1, 11):
    dff = df[df["naal_edition"] == x]
    dfg = df[df["naal_edition"] == x + 1]
    dff_works = set(dff["work_id"]) - set(dff["parent_id"])  # see above
    dfg_works = set(dfg["work_id"]) - set(dfg["parent_id"])
    dff_authors = set(dff["author_id"])
    dfg_authors = set(dfg["author_id"])

    # get output
    d = dict()
    d["naal_edition"] = x
    d["authors"] = len(dff_authors)
    d["works"] = len(dff_works)
    if x != 10:
        d["authors_reselected"] = len(dff_authors & dfg_authors)
        d["works_reselected"] = len(dff_works & dfg_works)
    l.append(d)
```

[12]:
```
%%capture cap --no-stderr
print(pd.DataFrame(l).to_markdown(index=False))
```

[13]:
```
display(Markdown(cap.stdout))
```

naal_edition	authors	works	authors_reselected	works_reselected
1	131	1160	121	809
2	155	1033	146	883
3	179	1228	169	1080
4	202	1316	183	928
5	224	1185	207	1041
6	239	1163	207	855
7	282	1248	253	1073
8	289	1218	245	977
9	294	1149	248	873
10	288	1025	nan	nan

2.4 Table 1 description

Number of authors and works selected for and across editions. On average, editors reselect 90% of authors from one edition to the next, whereas they reselect 80% of works.

[14]:
```
data = pd.DataFrame(l)
```

[15]:
```
mean_author_reselection = (data["authors_reselected"] / data["authors"]).mean()
print(f"mean author reselection: {round(mean_author_reselection * 100)}%")
```

```
mean author reselection: 90%
```

[16]:
```
mean_work_reselection = (data["works_reselected"] / data["works"]).mean()
print(f"mean work reselection: {round(mean_work_reselection * 100)}%")
```

```
mean work reselection: 80%
```

2.5 ...there are 103 authors who have been reselected for every edition.

[17]:
```
cols = ["author_id", "author_name", "naal_edition"]
author_count = (
    df[cols]
    .drop_duplicates()[["author_id", "author_name"]]
    .value_counts()
    .reset_index()
)
author_count.columns = ["author_id", "author_name", "count"]
author_count.sample(3)
```

[17]:

	author_id	author_name	count
150	274	Samson Occom	7

```
320        327  Jane Colman Turell      2
287        396      David L. Smith      3
```

[18]:
```
author_count[author_count["count"] == 10]["author_id"].unique().size
```

[18]:
```
103
```

[19]:
```
l = list(author_count[author_count["count"] == 10]["author_name"])
```

[20]:
```
# sort values by last name
l = sorted(l, key=lambda x: x.split()[-1])
```

[21]:
```
# this the string used to generate the list above
# some manual sorting was required for cases where the above method fails
# e.g., "Du Bois" should be under D not B

n103 = "; ".join(l)
```

2.6 They are overwhelmingly male (76%), even more overwhelmingly white (81%), and mostly both (61% are white men).

[22]:
```
n103 = author_count[author_count["count"] == 10]["author_id"].unique()
```

[23]:
```
cols = ["author_id", "gender_name", "race_name"]
df[df["author_id"].isin(n103)][cols].drop_duplicates()["gender_name"].
 ↪value_counts()[
    "Male"
] / len(n103)
```

[23]:
```
0.7572815533980582
```

[24]:
```
df[df["author_id"].isin(n103)][cols].drop_duplicates()["race_name"].
 ↪value_counts()[
    "White"
] / len(n103)
```

[24]:
```
0.8058252427184466
```

[25]:
```
df[(df["author_id"].isin(n103)) & (df["race_id"] == 10)][cols].
 ↪drop_duplicates()[
    ["gender_name", "race_name"]
].value_counts()
```

[25]:
```
gender_name  race_name
Male         White        63
Female       White        20
Name: count, dtype: int64
```

[26]:
```
63 / 103
```

[26]:
```
0.6116504854368932
```

2.7 *NAAL* authors in the tenth edition have fewer than half as many works (mean 3.6, median 2) as they did in the first edition (mean 8.9, median 5).

[27]:
```
l = list()

for author in df["author_id"].unique():
    for x in range(1, 11):
        dff = df[(df["naal_edition"] == x) & (df["author_id"] == author)]
        if len(dff) > 0:
            # output
            d = dict()
            d["naal_edition"] = x
            d["author_id"] = author
            d["works"] = len(
                set(dff["work_id"]) - set(dff["parent_id"])
            )  # works not also parents in a given edition
            l.append(d)
```

[28]:
```
works_per_edition = pd.DataFrame(l)
works_per_edition[["naal_edition", "works"]].groupby("naal_edition").median()
```

[28]:

naal_edition	works
1	5.0
2	3.0
3	3.0
4	3.0
5	3.0
6	3.0
7	2.0
8	2.0
9	2.0
10	2.0

[29]:
```
works_per_edition[["naal_edition", "works"]].groupby("naal_edition").mean()
```

[29]:

naal_edition	works
1	8.854962
2	6.664516
3	6.860335
4	6.514851
5	5.290179

```
6              4.866109
7              4.425532
8              4.217993
9              3.908163
10             3.559028
```

2.8 This covers 464 unique authors and 3,374 unique works across ten editions.

[30]:
```
df["author_id"].unique().size
```

[30]:
```
464
```

[31]:
```
df["work_id"].unique().size
```

[31]:
```
3374
```

[32]:
```
df["naal_edition"].unique().size
```

[32]:
```
10
```

2.9 In the tenth edition of the *NAAL*, 6.6% of the writers are Jewish

As stated in the paper, we use race and ethnicity categories from the 2020 Census, plus the designation Jewish to accurately account for *Jewish American Literature: A Norton Anthology* (2001).

Although this variable is called `race_name` here, we do not intend to imply that "Jewish" (or any of our other categories) can be unproblematically described as a "race" in practice. Note that the 2020 Census questionnaire itself says "Hispanic origins are not races."

For more on our treatment of race and ethnicity, including our selection of categories for the database, see the essay, especially endnote 12.

[33]:
```
df[(df["naal_edition"] == 10) & (df["race_name"] == "Jewish")][
    "author_id"
].unique().size / df[df["naal_edition"] == 10]["author_id"].unique().size
```

[33]:
```
0.06597222222222222
```

2.10 Authors tagged in our data as white in the most recent edition of the *NAAL* make up 57% of the whole. This is a substantial change from the 81% they took up in the first edition

[34]:
```
df[(df["naal_edition"] == 10) & (df["race_id"] == 10)]["author_id"].unique().
↪size / df[
    df["naal_edition"] == 10
]["author_id"].unique().size
```

[34]:
```
0.5729166666666666
```

[35]:
```
df[(df["naal_edition"] == 1) & (df["race_id"] == 10)]["author_id"].unique().
 ↪size / df[
    df["naal_edition"] == 1
]["author_id"].unique().size
```

[35]: 0.8091603053435115

2.11 The proportion of Black writers has also risen from 11% to 20%

[36]:
```
df[(df["naal_edition"] == 1) & (df["race_id"] == 4)]["author_id"].unique().size␣
 ↪/ df[
    df["naal_edition"] == 1
]["author_id"].unique().size
```

[36]: 0.10687022900763359

[37]:
```
df[(df["naal_edition"] == 10) & (df["race_id"] == 4)]["author_id"].unique().
 ↪size / df[
    df["naal_edition"] == 10
]["author_id"].unique().size
```

[37]: 0.2013888888888889

2.12 Indigenous writers are, by the standards of contemporary demography, overrepresented in the anthology, making up about 9% of authors

[38]:
```
df[(df["naal_edition"] == 10) & (df["race_id"] == 8)]["author_id"].unique().
 ↪size / df[
    df["naal_edition"] == 10
]["author_id"].unique().size
```

[38]: 0.09027777777777778

2.13 Just over 2% of tenth edition *NAAL* authors are Asian American or Pacific Islanders, while this group makes up about 7% of the US population. For Latina/o/x writers, those numbers are about 4% and 19%, respectively.

[39]:
```
df[["race_id", "race_name"]].drop_duplicates().dropna().sort_values("race_id")
```

[39]:

	race_id	race_name
1387	3.0	Other Asian
214	4.0	Black or African American
7	6.0	Hispanic, Latino, or Spanish origin
33	8.0	American Indian or Alaska Native
1	10.0	White
633	11.0	Jewish
698	12.0	Chinese

```
2466      13.0                        Filipino
1419      14.0                   Asian Indian
13364     15.0                     Vietnamese
2813      17.0                        Japanese
```

[40]:
```
aapi = [3, 12, 13, 14, 15, 17]
```

[41]:
```
# AAPI
df[(df["naal_edition"] == 10) & (df["race_id"].isin(aapi))][
    "author_id"
].unique().size / df[df["naal_edition"] == 10]["author_id"].unique().size
```

[41]:
```
0.024305555555555556
```

[42]:
```
# Hispanic, Latino, or Spanish origin
df[(df["naal_edition"] == 10) & (df["race_id"] == 6)]["author_id"].unique().
 ↪size / df[
    df["naal_edition"] == 10
]["author_id"].unique().size
```

[42]:
```
0.041666666666666664
```

2.14 If these three were marked "white," that would leave only 3% of tenth edition authors who were Latina/o/x.

[43]:
```
l10 = df[(df["naal_edition"] == 10) & (df["race_id"] == 6)]["author_id"].
 ↪unique().size
l10
```

[43]:
```
12
```

[44]:
```
(l10 - 3) / df[df["naal_edition"] == 10]["author_id"].unique().size
```

[44]:
```
0.03125
```

2.15 In 1979, women made up 22% of anthologized authors. In 2022, that figure has grown to 36%. ...the eighth (34%) and ninth (30%) editions each had *declining* relative female authorship, before a rebound in the tenth.

[45]:
```
l = list()

for x in range(1, 11):
    dff = df[df["naal_edition"] == x]
    d = dict()
    d["naal_edition"] = x
    d["authors"] = dff["author_id"].unique().size
    d["women"] = dff[dff["gender_id"] == 2]["author_id"].unique().size
```

```
    d["% women"] = d["women"] / d["authors"]
    l.append(d)
```

[46]:
```
%%capture cap --no-stderr
print(pd.DataFrame(l).to_markdown(index=False))
```

[47]:
```
display(Markdown(cap.stdout))
```

naal_edition	authors	women	% women
1	131	29	0.221374
2	155	35	0.225806
3	179	49	0.273743
4	202	53	0.262376
5	224	72	0.321429
6	239	79	0.330544
7	282	98	0.347518
8	289	98	0.3391
9	294	89	0.302721
10	288	105	0.364583

2.16 ...in the latest *NAAL*, just 39% of works were written by women

[48]:
```
l = list()

for x in range(1, 11):
    dff = df[df["naal_edition"] == x]
    works = set(dff["work_id"]) - set(dff["parent_id"])
    d = dict()
    d["naal_edition"] = x
    d["works"] = len(works)
    d["works by women"] = (
        dff[(dff["work_id"].isin(works)) & (dff["gender_id"] == 2)]["work_id"]
        .unique()
        .size
    )
    d["% works by women"] = d["works by women"] / d["works"]
    l.append(d)
```

[49]:
```
%%capture cap --no-stderr
print(pd.DataFrame(l).to_markdown(index=False))
```

[50]:
```
display(Markdown(cap.stdout))
```

naal_edition	works	works by women	% works by women
1	1160	246	0.212069

naal_edition	works	works by women	% works by women
2	1033	236	0.228461
3	1228	307	0.25
4	1316	366	0.278116
5	1185	405	0.341772
6	1163	395	0.339639
7	1248	439	0.351763
8	1218	441	0.362069
9	1149	408	0.355091
10	1025	395	0.385366

2.17 ...and Emily Dickinson accounts for more than 18% of that

[51]:
```
dickinson_id = df[df["author_name"].str.contains("Emily Dickinson")][
    "author_id"
].unique()[0]
dickinson_id
```

[51]:
```
382
```

[52]:
```
dickinson10 = len(
    set(df[(df["author_id"] == dickinson_id) & (df["anthology_id"] ==␣
 ↪10)]["work_id"])
    - set(df[df["anthology_id"] == 10]["parent_id"])
)
```

[53]:
```
women_works10 = len(
    set(df[(df["anthology_id"] == 10) & (df["gender_id"] == 2)]["work_id"])
    - set(df[df["anthology_id"] == 10]["parent_id"])
)
```

[54]:
```
dickinson10 / women_works10
```

[54]:
```
0.18137254901960784
```

2.18 At the average pace set by the first ten editions, women authors will not reach parity with men until the nineteenth edition of the *NAAL*, which will feature 445 authors and be published around 2065.

[55]:
```
l = list()

for i, x in enumerate(range(1, 11)):
    dff = df[df["naal_edition"] == x]
    d = dict()
    d["naal_edition"] = x
    d["authors"] = dff["author_id"].unique().size
```

```
    d["men"] = dff[dff["gender_id"] == 1]["author_id"].unique().size
    d["women"] = dff[dff["gender_id"] == 2]["author_id"].unique().size
    d["% women"] = d["women"] / d["authors"]
    if x > 1:
        dfg = dff = df[df["naal_edition"] == x - 1]
        d["author change"] = d["authors"] - dfg["author_id"].unique().size
        d["men change"] = (
            d["men"] - dfg[dfg["gender_id"] == 1]["author_id"].unique().size
        )
        d["women change"] = (
            d["women"] - dfg[dfg["gender_id"] == 2]["author_id"].unique().size
        )

    l.append(d)
```

[56]:
```
authors_pace = pd.DataFrame(l)
```

[57]:
```
authors_pace["% women change"] = (
    authors_pace["% women"] - authors_pace["% women"].shift()
)
```

[58]:
```
%%capture cap --no-stderr
print(authors_pace.to_markdown(index=False))
```

[59]:
```
display(Markdown(cap.stdout))
```

naal_edition	authors	men	women	% women	author change	men change	women change	% women change
1	131	102	29	0.221374	nan	nan	nan	nan
2	155	120	35	0.225806	24	18	6	0.00443241
3	179	129	49	0.273743	24	9	14	0.0479366
4	202	145	53	0.262376	23	16	4	-0.0113668
5	224	145	72	0.321429	22	0	19	0.0590523
6	239	152	79	0.330544	15	7	7	0.00911536
7	282	176	98	0.347518	43	24	19	0.0169738
8	289	186	98	0.3391	7	10	0	-0.00841738
9	294	201	89	0.302721	5	15	-9	-0.0363793
10	288	179	105	0.364583	-6	-22	16	0.0618622

[60]:
```
avg_pct_women_change = authors_pace["% women change"].mean()
avg_author_change = authors_pace["author change"].mean()
avg_women_change = authors_pace["women change"].mean()
avg_men_change = authors_pace["men change"].mean()
```

[61]:
```
# how many editions remain until parity?
remaining_eds = (0.5 - authors_pace.iloc[9, 4]) / avg_pct_women_change
```

```
remaining_eds
```

[61]: 8.510272071071629

[62]:
```
# average time between editions
years = [
    1979,
    1985,
    1989,
    1994,
    1998,
    2003,
    2007,
    2012,
    2017,
    2022,
]  # naal edition pub years
avg_time_between_eds = pd.Series([x - y for x, y in zip(years[1:], years)]).
 ↪mean()
```

[63]:
```
# round up because parity reached between 8 and 9 more editions
(round(remaining_eds) * avg_time_between_eds) + 2022
```

[63]: 2065.0

[64]:
```
# author growth over next 9 editions; round to a whole author
(avg_author_change * round(remaining_eds)) + authors_pace.iloc[9, 1]
```

[64]: 445.0

2.19 The *NAAL* has gotten longer since 1979, increasing its page count by about 17%.

[65]:
```
# data of pages per naal edition formatted as csv
pages_ed = """1,4982
2,5175
3,5295
4,5400
5,5510
6,5696
7,5846
8,6000
9,6107
10,5840"""
```

[66]:
```
pages_ed = pd.DataFrame(
    [row.split(",") for row in pages_ed.split("\n")], columns=["naal_edition",␣
 ↪"pages"]
```

```
)
```

[67]:
```
for col in pages_ed.columns:
    pages_ed[col] = pd.to_numeric(pages_ed[col])
```

[68]:
```
pages_ed.set_index("naal_edition", inplace=True)
```

[69]:
```
pages_ed["authors"] = (
    df[["naal_edition", "author_id"]]
    .drop_duplicates()
    .groupby("naal_edition")
    .count()
    .values
)
```

[70]:
```
pages_ed
```

[70]:

naal_edition	pages	authors
1	4982	131
2	5175	155
3	5295	179
4	5400	202
5	5510	224
6	5696	239
7	5846	282
8	6000	289
9	6107	294
10	5840	288

[71]:
```
pages_ed.iloc[9, 0] / pages_ed.iloc[0, 0]
```

[71]:
```
1.172219991971096
```

2.20 If the ratio of the number of authors to the number of pages in the first edition of the anthology had been maintained, there would be about 154 authors in the current edition. But the current *NAAL* has 288 authors.

[72]:
```
pp_author = pages_ed.iloc[0, 0] / pages_ed.iloc[0, 1]
pages_ed.iloc[9, 0] / pp_author
```

[72]:
```
153.56081894821358
```

[73]:
```
df[df["naal_edition"] == 10]["author_id"].unique().size
```

[73]:
```
288
```

2.21 While the current edition contains 0.88 times as many works as appear in the first edition, it contains 2.20 times the number of authors.

[74]:
```
naal1_works = len(
    set(df[df["naal_edition"] == 1]["work_id"])
    - set(df[df["naal_edition"] == 1]["parent_id"])
)
naal10_works = len(
    set(df[df["naal_edition"] == 10]["work_id"])
    - set(df[df["naal_edition"] == 10]["parent_id"])
)
```

[75]:
```
print(f"1:\t{naal1_works}\n10:\t{naal10_works}")
```

```
1:      1160
10:     1025
```

[76]:
```
naal10_works / naal1_works
```

[76]: 0.8836206896551724

[77]:
```
naal1_authors = df[df["naal_edition"] == 1]["author_id"].unique().size
naal10_authors = df[df["naal_edition"] == 10]["author_id"].unique().size
```

[78]:
```
naal10_authors
```

[78]: 288

[79]:
```
naal10_authors / naal1_authors
```

[79]: 2.198473282442748

2.22 On average, 89% of authors who appear in any given edition will be reselected for the following one.

[80]:
```
l = list()

for x in range(1, 10):  # skip 10 since no reselection opportunity yet
    d = dict()
    d["naal_edition"] = x
    authors = df[df["naal_edition"] == x]["author_id"].unique()
    next_authors = df[df["naal_edition"] == x + 1]["author_id"].unique()
    d["authors"] = authors.size
    d["next_authors"] = next_authors.size
    d["reselections"] = len(set(authors) & set(next_authors))
    l.append(d)
```

[81]:
```
reselects = pd.DataFrame(l)
reselects["reselect %"] = reselects["reselections"].divide(reselects["authors"])
reselects["reselect % of next ed"] = reselects["reselections"].divide(
    reselects["next_authors"]
)
```

[82]:
```
%%capture cap --no-stderr
print(reselects.to_markdown(index=False))
```

[83]:
```
display(Markdown(cap.stdout))
```

naal_edition	authors	next_authors	reselections	reselect %	reselect % of next ed
1	131	155	121	0.923664	0.780645
2	155	179	146	0.941935	0.815642
3	179	202	169	0.944134	0.836634
4	202	224	183	0.905941	0.816964
5	224	239	207	0.924107	0.866109
6	239	282	207	0.866109	0.734043
7	282	289	253	0.897163	0.875433
8	289	294	245	0.847751	0.833333
9	294	288	248	0.843537	0.861111

[84]:
```
# reselections / opportunities
reselects["reselections"].sum() / reselects["authors"].sum()
```

[84]: 0.8917293233082707

2.23 Slightly more than half (234 or 50.4%) of all authors ever selected have been reselected for every revision following their initial selection.

[85]:
```
cols = ["author_id", "naal_edition"]
l = list()

for author in df["author_id"].unique():
    dff = df[df["author_id"] == author][cols].drop_duplicates()
    appearances = dff.shape[0]
    max_edition = dff["naal_edition"].max()
    min_edition = dff["naal_edition"].min()

    if max_edition == 10:
        opportunities = appearances - 1
    else:
        opportunities = appearances

    rs = 0
    editions = list(dff["naal_edition"])
```

```
    for ed in editions:
        if (ed + 1) in editions:
            rs += 1

    # output
    d = dict()
    d["author id"] = author
    d["min NAAL edition"] = min_edition
    d["max NAAL edition"] = max_edition
    d["total appearances"] = appearances
    d["opportunities for reselection"] = opportunities
    d["consecutive reselections"] = rs
    if opportunities > 0:
        d["reselection rate"] = rs / opportunities
    else:
        d["reselection rate"] = None
    l.append(d)
```

[86]:
```
survival = pd.DataFrame(l)
```

[87]:
```
# proportion of authors who have survived all subsequent revisions
survival[survival["reselection rate"] == 1].shape[0] / survival.shape[0]
```

[87]:
```
0.5043103448275862
```

[88]:
```
survival[survival["reselection rate"] == 1]["consecutive reselections"].
↪describe()
```

[88]:
```
count    234.000000
mean       6.290598
std        2.967810
min        1.000000
25%        3.000000
50%        8.000000
75%        9.000000
max        9.000000
Name: consecutive reselections, dtype: float64
```

2.24 Although white men have always been the most-cut group...

n.b., In the seventh edition, they tie with white women.

[89]:
```
l = list()

for x in range(1, 10):
    dff = df[df["naal_edition"] == x][
        ["author_id", "naal_edition", "race_name", "gender_name"]
    ].drop_duplicates()
```

```
    dfg = df[df["naal_edition"] == x + 1]
    cuts = set(dff["author_id"]) - set(dfg["author_id"])
    cuts = dff[dff["author_id"].isin(cuts)]
    cuts = cuts.groupby(["naal_edition", "gender_name", "race_name"]).count()
    cuts.columns = ["#cut"]
    cuts["max"] = cuts["#cut"].max()
    l.append(cuts)
```

[90]:
```
cuts_ed = pd.concat(l).reset_index()
cuts_ed["is_max"] = cuts_ed["#cut"] == cuts_ed["max"]
```

[91]:
```
%%capture cap --no-stderr
print(cuts_ed[cuts_ed["is_max"] == True].to_markdown(index=False))
```

[92]:
```
display(Markdown(cap.stdout))
```

naal_edition	gender_name	race_name	#cut	max	is_max
1	Male	White	5	5	True
2	Male	White	6	6	True
3	Male	White	7	7	True
4	Male	White	12	12	True
5	Male	White	10	10	True
6	Male	White	12	12	True
7	Female	White	9	9	True
7	Male	White	9	9	True
8	Male	White	17	17	True
9	Male	White	24	24	True

2.25 ...they have always been the largest group of writers, peaking in the second edition when they held nearly two-thirds (100 of 155) of all author spots

[93]:
```
l = list()

for x in range(1, 11):
    dff = df[df["naal_edition"] == x].drop_duplicates()
    d = dict()
    d["naal_edition"] = x
    d["authors"] = dff["author_id"].unique().size
    d["white men"] = (
        dff[(dff["gender_id"] == 1) & (dff["race_id"] == 10)]["author_id"].
 ↪unique().size
    )
    l.append(d)
```

[94]:
```
w = pd.DataFrame(l)
```

[95]:
```
w["% white men"] = w["white men"].divide(w["authors"])
```

[96]:
```
%%capture cap --no-stderr
print(w.to_markdown(index=False))
```

[97]:
```
display(Markdown(cap.stdout))
```

naal_edition	authors	white men	% white men
1	131	84	0.641221
2	155	100	0.645161
3	179	104	0.581006
4	202	103	0.509901
5	224	100	0.446429
6	239	101	0.422594
7	282	115	0.407801
8	289	121	0.418685
9	294	130	0.442177
10	288	112	0.388889

2.26 The increase in white male authors over time (up from 84 to 112 authors, a gain of 28 spots) surpasses the total number of Black women authors anthologized today (25). Among authorial groups aggregated on the intersection of their race and gender, only white women (51) and Black men (33) now have more spots *in total* than white male authors have *gained* since the first edition.

[98]:
```
# white men authors increase
w.iloc[9, 2] - w.iloc[0, 2]
```

[98]: 28

[99]:
```
%%capture cap --no-stderr
cols = ["author_id", "naal_edition", "race_name", "gender_name"]
print(df[df["naal_edition"] == 10][cols].drop_duplicates().
 ↪groupby("naal_edition")[
    ["race_name", "gender_name"]
].value_counts().reset_index().to_markdown(index=False))
```

[100]:
```
display(Markdown(cap.stdout))
```

naal_edition	race_name	gender_name	count
10	White	Male	112
10	White	Female	51
10	Black or African American	Male	33
10	Black or African American	Female	25

naal_edition	race_name	gender_name	count
10	American Indian or Alaska Native	Male	16
10	Jewish	Female	11
10	American Indian or Alaska Native	Female	9
10	Hispanic, Latino, or Spanish origin	Male	8
10	Jewish	Male	8
10	Hispanic, Latino, or Spanish origin	Female	4
10	Chinese	Female	3
10	Asian Indian	Female	1
10	Chinese	Male	1
10	Japanese	Female	1
10	Vietnamese	Male	1

2.27 This holds true even though the tenth edition deaccessioned 24 white male writers, the largest such cut in the anthology's history

[101]:
```
l = list()

for x in range(1, 11):
    dff = df[(df["naal_edition"] == x) & (df["gender_id"] == 1) &␣
 ↪(df["race_id"] == 10)]

    dfg = df[
        (df["naal_edition"] == x + 1) & (df["gender_id"] == 1) & (df["race_id"]␣
 ↪== 10)
    ]

    d = dict()
    d["naal_edition"] = x
    d["white men"] = dff["author_id"].unique().size
    if x < 10:
        d["white men cut"] = len(
            set(dff["author_id"].unique()) - set(dfg["author_id"].unique())
        )
    l.append(d)
```

[102]:
```
%%capture cap --no-stderr
print(pd.DataFrame(l).to_markdown(index=False))
```

[103]:
```
display(Markdown(cap.stdout))
```

naal_edition	white men	white men cut
1	84	5
2	100	6
3	104	7

naal_edition	white men	white men cut
4	103	12
5	100	10
6	101	12
7	115	9
8	121	17
9	130	24
10	112	nan

2.28 Yet, of the 333 authors added to the anthology since the first edition, only 101 of them were born after the youngest author in the first edition (Amiri Baraka, born 1934).

[104]:
```
df[df["naal_edition"] == 1]["birth_year"].max()
```

[104]:
```
1934.0
```

[105]:
```
naal1 = df[df["naal_edition"] == 1]["author_id"].unique()
added_authors = df[df["naal_edition"] > 1]["author_id"].unique()
len(set(added_authors) - set(naal1))
```

[105]:
```
333
```

[106]:
```
(
    df[(df["author_id"].isin(added_authors)) & (df["birth_year"] > 1934)][
        ["author_id", "author_name", "birth_year"]
    ]
    .drop_duplicates()
    .sort_values("birth_year")
    .shape[0]
    + 2 # because two authors were born after Baraka in 1934: Joan Didion and␣
 ↪Gerald Vizenor.
)
```

[106]:
```
101
```

2.29 ...we see a more stochastic pattern, with a median birth year of 1877 in the first edition, the highest in the sixth (1888), and the lowest in the seventh, ninth and tenth editions (all 1876).

[107]:
```
l = list()

for x in range(1, 11):
    dff = df[df["naal_edition"] == x][["author_id", "birth_year"]].
 ↪drop_duplicates()
    d = dict()
```

```
    d["naal_edition"] = x
    d["authors"] = dff["author_id"].unique().size
    d["min"] = dff["birth_year"].min()
    d["max"] = dff["birth_year"].max()
    d["mean"] = dff["birth_year"].mean()
    d["median"] = dff["birth_year"].median()
    l.append(d)
```

[108]:
```
%%capture cap --no-stderr
print(pd.DataFrame(l).set_index("naal_edition").to_markdown())
```

[109]:
```
display(Markdown(cap.stdout))
```

naal_edition	authors	min	max	mean	median
1	131	1588	1934	1851.59	1877
2	155	1579	1944	1851.05	1885
3	179	1579	1955	1855.9	1887
4	202	1451	1957	1839.99	1879
5	224	1451	1959	1845.96	1887
6	239	1451	1963	1849.13	1887.5
7	282	1451	1967	1853.66	1876
8	289	1451	1978	1855.89	1880.5
9	294	1451	1972	1848.99	1876
10	288	1451	1986	1851.82	1876

2.30 In four of nine revisions, editors selected more new authors at or below the median birth year than above it.

[110]:
```
l = list()

for x in range(2, 11):
    dff = df[df["naal_edition"] == x][["author_id", "birth_year"]].
 ↪drop_duplicates()
    dfg = df[df["naal_edition"] == x - 1][
        ["author_id", "birth_year"]
    ].drop_duplicates()  # get previous ed
    d = dict()
    d["naal edition"] = x
    d["authors"] = dff["author_id"].size
    d["authors previous"] = dfg["author_id"].size
    d["median birth"] = dff["birth_year"].median()
    d["reselected from previous"] = len(set(dff["author_id"]) &␣
 ↪set(dfg["author_id"]))
    # new selections
    new = set(dff["author_id"]) - set(dfg["author_id"])
    d["newly selected"] = len(new)
```

```
    d["new at or below median"] = dff[
        (dff["birth_year"] <= d["median birth"]) & (dff["author_id"].isin(new))
    ].shape[0]
    d["new above median"] = dff[
        (dff["birth_year"] > d["median birth"]) & (dff["author_id"].isin(new))
    ].shape[0]
    l.append(d)
```

[111]:
```
%%capture cap --no-stderr
print(pd.DataFrame(l).to_markdown(index=False))
```

[112]:
```
display(Markdown(cap.stdout))
```

naal edition	authors	authors previous	median birth	reselected from previous	newly selected	new at or below median	new above median
2	155	131	1885	121	34	11	23
3	179	155	1887	146	33	8	24
4	202	179	1879	169	33	20	10
5	224	202	1887	183	41	12	26
6	239	224	1887.5	207	32	16	13
7	282	239	1876	207	75	47	28
8	289	282	1880.5	253	36	12	23
9	294	289	1876	245	49	27	16
10	288	294	1876	248	40	12	25

[113]:
```
new = pd.DataFrame(l)
(new["new at or below median"] > new["new above median"]).value_counts()
```

[113]:
```
False    5
True     4
Name: count, dtype: int64
```

2.31 The total number of poets (defined as authors with at least one poem in a given edition) in each edition grew rapidly through the seventh edition, when it peaked at 114, but it has fallen ever since, landing at 88 in the tenth edition. The total number of poems has fallen, too, from 829 in the third edition to 525 in the tenth.

[114]:
```
# add form data
forms = "forms.csv"
forms = pd.read_csv(forms)[["work id", "form"]]
forms.sample()
```

[114]:
```
     work id    form
839     2720  poetry
```

[115]:
```
df = df.merge(forms, left_on="work_id", right_on="work id")
```

[116]:
```
poets = df[df["form"] == "poetry"]["author_id"].unique()
```

[117]:
```
l = list()

for x in range(1, 11):
    dff = df[df["naal_edition"] == x]
    works = set(dff["work_id"]) - set(dff["parent_id"])
    poets = dff[(dff["work_id"].isin(works)) & (dff["form"] == "poetry")][
        "author_id"
    ].unique()
    poems = dff[(dff["work_id"].isin(works)) & (dff["form"] == "poetry")][
        "work_id"
    ].unique()

    # output
    d = dict()
    d["naal_edition"] = x
    d["authors"] = dff["author_id"].unique().size
    d["works"] = len(works)
    d["poets"] = poets.size
    d["poems"] = poems.size
    l.append(d)
```

[118]:
```
poetry = pd.DataFrame(l)
```

[119]:
```
%%capture cap --no-stderr
print(poetry.to_markdown(index=False))
```

[120]:
```
display(Markdown(cap.stdout))
```

naal_edition	authors	works	poets	poems
1	131	1160	61	777
2	155	1033	71	683
3	179	1228	77	766
4	202	1316	85	829
5	224	1185	87	762
6	239	1163	97	748
7	282	1248	114	752
8	289	1218	107	707
9	294	1149	93	637
10	288	1025	88	525

2.32 Poets held 47% of author spots in the first edition and hold 31% today.

[121]:
```
poetry["% poets"] = poetry["poets"].divide(poetry["authors"])
poetry["% poems"] = poetry["poems"].divide(poetry["works"])
```

[122]:
```
%%capture cap --no-stderr
print(poetry.to_markdown(index=False))
```

[123]:
```
display(Markdown(cap.stdout))
```

naal_edition	authors	works	poets	poems	% poets	% poems
1	131	1160	61	777	0.465649	0.669828
2	155	1033	71	683	0.458065	0.661181
3	179	1228	77	766	0.430168	0.623779
4	202	1316	85	829	0.420792	0.629939
5	224	1185	87	762	0.388393	0.643038
6	239	1163	97	748	0.405858	0.643164
7	282	1248	114	752	0.404255	0.602564
8	289	1218	107	707	0.370242	0.58046
9	294	1149	93	637	0.316327	0.554395
10	288	1025	88	525	0.305556	0.512195

2.33 In the first edition, poets had a median of 10 poems. This has dropped to just 4.5 in the tenth edition.

[124]:
```
l = list()

for poet in poets:
    for x in range(1, 11):
        dff = df[(df["author_id"] == poet) & (df["naal_edition"] == x)]
        if len(dff) > 0:  # if they have a poem in an edition
            # output
            d = dict()
            d["poet_id"] = poet
            d["naal_edition"] = x
            d["works"] = dff["work_id"].unique().size
            d["poems"] = dff[dff["form"] == "poetry"]["work_id"].unique().size
            if x != 10:
                next_ed = df[(df["naal_edition"] == x + 1) & (df["author_id"]␣
 ↪== poet)]
                d["works reselected"] = len(
                    set(dff["work_id"].unique()) & set(next_ed["work_id"].
 ↪unique())
                )
                d["poems reselected"] = len(
                    set(dff[dff["form"] == "poetry"]["work_id"].unique())
```

```
                    & set(next_ed[next_ed["form"] == "poetry"]["work_id"].
 ↪unique())
                )
            l.append(d)
```

[125]:
```
poetry = pd.DataFrame(l)
```

[126]:
```
poetry.sample()
```

[126]:

	poet_id	naal_edition	works	poems	works reselected	poems reselected
294	290	9	6	6	6.0	6.0

[127]:
```
%%capture cap --no-stderr
# median works and poems per edition
cols = ["naal_edition", "works", "poems", "works reselected", "poems␣
 ↪reselected"]
print(poetry[cols].groupby("naal_edition").median().reset_index().
 ↪to_markdown(index=False))
```

[128]:
```
display(Markdown(cap.stdout))
```

naal_edition	works	poems	works reselected	poems reselected
1	12	11	9	8
2	9	9	7.5	7.5
3	9	9	8	8
4	10	9	8	7
5	8	7	7	7
6	7	7	6	6
7	7	7	6	6
8	7	6	6	6
9	6	6	5	5
10	5	4.5	nan	nan

2.34 There is no better evidence for this than the 73 times that editors cut all of an author's previously anthologized works, then reselected that author with entirely new works.

[129]:
```
l = list()

for x in range(1, 10):  # skip last
    dff = df[df["naal_edition"] == x]
    dfg = df[df["naal_edition"] == x + 1]
    reselected = set(dff["author_id"]) & set(dfg["author_id"])
    for author in reselected:
        e1_works = set(dff[dff["author_id"] == author]["work_id"])
        e2_works = set(dfg[dfg["author_id"] == author]["work_id"])
```

```
        overlap = e1_works & e2_works
        d = dict()
        d["naal_edition"] = x
        d["naal_edition_compared"] = x + 1
        d["author_id"] = author
        d["naal_e1_works"] = len(e1_works)
        d["naal_e2_works"] = len(e2_works)
        d["overlap"] = len(overlap)
        l.append(d)
```

[130]:
```
overlap = pd.DataFrame(l).drop_duplicates()
```

[131]:
```
overlap[(overlap["overlap"] == 0) & (overlap["naal_e2_works"] > 0)].shape[0]
```

[131]: 73

2.35 Only 9% of the texts selected in the history of the *NAAL* (314 of 3,374) have been reselected for every edition.

[132]:
```
works_ed = df[["work_id", "naal_edition"]].groupby("work_id").count().
↪reset_index()
```

[133]:
```
works_ed.columns = ["work_id", "#editions"]
works_ed.sample(3)
```

[133]:

	work_id	#editions
113	1716	8
2457	4863	4
97	1700	3

[134]:
```
total_works = df["work_id"].unique().size
total_works
```

[134]: 3374

[135]:
```
works_ed[works_ed["#editions"] == 10].shape[0]
```

[135]: 314

[136]:
```
works_ed[works_ed["#editions"] == 10].shape[0] / total_works
```

[136]: 0.09306461173681091

[137]:
```
# among works never parents
never_parents = set(df["work_id"]) - set(df["parent_id"])
works_ed = (
    df[df["work_id"].isin(never_parents)][["work_id", "naal_edition"]]
    .groupby("work_id")
```

```
    .count()
    .reset_index()
)
works_ed.columns = ["work_id", "#editions"]
works_ed[works_ed["#editions"] == 10].shape[0] / len(never_parents)
```

[137]: 0.09441166261714683

2.36 Since 83% of the women and people of color who have ever been in the anthology were added after the first edition through the strategy of authorial growth...

[138]:
```
# total
wm = df[(df["race_id"] == 10) & (df["gender_id"] == 1)]["author_id"].unique()
not_wm = set(df["author_id"]) - set(wm)
nwm = df[df["author_id"].isin(not_wm)]["author_id"].unique()
nwm1 = df[(df["author_id"].isin(not_wm)) & (df["naal_edition"] == 1)][
    "author_id"
].unique()
```

[139]:
```
len(set(nwm) - set(nwm1)) / len(set(nwm))
```

[139]: 0.8252788104089219

[140]:
```
# per edition
l = list()

for x in range(1, 11):
    dff = df[df["naal_edition"] == x]
    # output
    d = dict()
    d["naal_edition"] = x
    d["authors"] = dff["author_id"].unique().size
    wm = dff[(dff["race_id"] == 10) & (dff["gender_id"] == 1)]["author_id"].
 ↪unique()
    d["white men"] = wm.size
    d["not white men"] = dff[~dff["author_id"].isin(wm)]["author_id"].unique().
 ↪size
    l.append(d)
```

[141]:
```
data = pd.DataFrame(l)
data["diff"] = data["white men"] - data["not white men"]
```

[142]:
```
%%capture cap --no-stderr
print(data.to_markdown(index=False))
```

[143]:
```
display(Markdown(cap.stdout))
```

naal_edition	authors	white men	not white men	diff
1	131	84	47	37
2	155	100	55	45
3	179	104	75	29
4	202	103	99	4
5	224	100	124	-24
6	239	101	138	-37
7	282	115	167	-52
8	289	121	168	-47
9	294	130	164	-34
10	288	112	176	-64

2.37 This group represents an unbroken continuity with 1979, with its 63 white men overwhelming its women (25) and people of color (13 African Americans, zero indigenous people, zero Asian Americans or Pacific Islanders, and William Carlos Williams the only Latino person).

[144]:
```
cols = ["author_id", "author_name", "naal_edition"]
author_count = (
    df[cols]
    .drop_duplicates()[["author_id", "author_name"]]
    .value_counts()
    .reset_index()
)
author_count.columns = ["author_id", "author_name", "count"]
author_count.sample(3)
```

[144]:
```
     author_id        author_name  count
57         419      Edith Wharton     10
458        687  Hendrick Aupaumut      1
317        393           Leo Marx      2
```

[145]:
```
n103 = author_count[author_count["count"] == 10]["author_id"]
```

[146]:
```
cols = ["author_id", "gender_name"]
df[df["author_id"].isin(n103)][cols].drop_duplicates().groupby(
    "gender_name"
).count().reset_index()
```

[146]:
```
  gender_name  author_id
0      Female         25
1        Male         78
```

[147]:
```
cols = ["author_id", "race_name"]
df[df["author_id"].isin(n103)][cols].drop_duplicates().groupby(
    "race_name"
```

```
).count().reset_index().sort_values("author_id")
```

[147]:

	race_name	author_id
1	Hispanic, Latino, or Spanish origin	1
2	Jewish	6
0	Black or African American	13
3	White	83

[148]:
```
cols = ["author_id", "race_name", "gender_name"]
df[df["author_id"].isin(n103)][cols].drop_duplicates().groupby(
    ["gender_name", "race_name"]
).count().reset_index().sort_values("author_id")
```

[148]:

	gender_name	race_name	author_id
4	Male	Hispanic, Latino, or Spanish origin	1
0	Female	Black or African American	2
1	Female	Jewish	3
5	Male	Jewish	3
3	Male	Black or African American	11
2	Female	White	20
6	Male	White	63