CALIFORNIA GRANTS

EDA

Joshua Susanto
Hack for LA

Goal: To explain findings and progress in an understandable and accessible way

BACKGROUND INFORMATION

- Data came from the ca.gov public data portal
- Due to the The Grant Information Act of 2018, the State Library was required to build a website by July 1, 2020, "that provides a centralized location … to find state grant opportunities."
- All agencies were then required to post this information onto grants.ca.gov for the State Library.
- This data includes how to apply, links for more details, total amount given, etc. and is regularly updated
- Included was a dictionary for the columns

READING IN THE DATA

```
[2]: # import necessary packages
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

```
[3]: grants_raw = pd.read_csv("ca_grants.csv")
```

```
[4]: grants_raw.head()
```

[4]:

	PortalID	GrantID	Status	LastUpdated	ChangeNotes	AgencyDept	Title	Type	LOI	Categories	...	ApplicationDeadline	AwardPeriod	ExpAwardDate	ElecSubmissio
0	6481	DOJ-PROP56-2022-23-1	active	2022-07-18 17:28:54	Updated eligibility, suggested activities, and...	Department of Justice (Office of the Attorney ...	Tobacco Grant Program FY 2022-23 Request for P...	Grant	No	Education; Law, Justice, and Legal Services	...	2022-08-17 23:59:00	2-3 Years	October 2022	email: TobaccoGrantRFP@doj.ca.go
1	11966	NaN	active	2022-07-15 22:20:56	Application open date: July 15, 2022. Updated...	Department of Health Care Access and Information	California State Loan Repayment Program (SLRP)	Grant	No	Health & Human Services	...	2022-09-15 15:00:00	2 months	December 2022	url: https://funding.hcai.ca.gov
2	11960	NaN	active	2022-07-14 22:35:54	NaN	CA Arts Council	Administering Organization – Individual Artist...	Grant	No	Disadvantaged Communities; Libraries and Arts	...	2022-08-31 23:59:00	1/1/23 - 12/31/24	November 2022	url: https://calartscouncil.smartsimple.com/s_
3	11957	NaN	active	2022-07-14 22:19:54	NaN	CA Arts Council	Administering Organization – Arts Administrato...	Grant	No	Disadvantaged Communities; Education; Employme...	...	2022-08-31 23:59:00	1/1/23 - 12/31/24	November 2022	url: https://calartscouncil.smartsimple.com/s_
4	11912	NaN	active	2022-07-14 17:13:34	NaN	Department of Pesticide Regulation	Department of Pesticide Regulation 2023 Allian...	Grant	No	Agriculture; Disadvantaged Communities; Educat...	...	2022-12-08 23:59:00	2.5 years	Late June 2023	email: DPRpmGrants.solicitation@cdpr.ca.go

5 rows × 36 columns

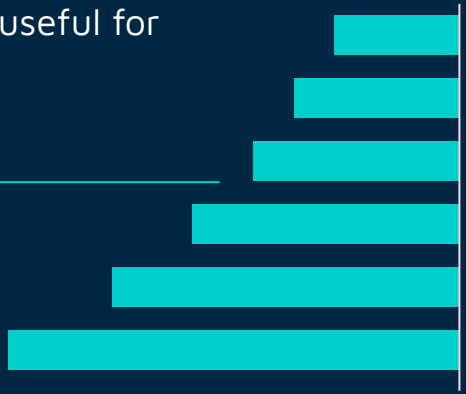
In order to properly read and start analyzing the data, our first step is to import necessary packages. We need the Pandas (and NumPy) package to read and analyze the data as well as Seaborn and matplotlib for data visualization.

Looking at our columns we notice columns that need to be manipulated or removed altogether.

REMOVING UNNECESSARY VARIABLES

We will remove columns...

- With excessive missing values
- With redundant information
- That cannot be realistically useful for our analysis



```
[8]: # Columns we are left with
     print(grants.info())

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 725 entries, 0 to 724
Data columns (total 21 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   PortalID             725 non-null    object
 1   Status               725 non-null    object
 2   LastUpdated          725 non-null    object
 3   AgencyDept           725 non-null    object
 4   Title                725 non-null    object
 5   Type                 725 non-null    object
 6   LOI                  724 non-null    object
 7   Categories           725 non-null    object
 8   Purpose              724 non-null    object
 9   Description          725 non-null    object
 10  ApplicantType        721 non-null    object
 11  FundingSource        721 non-null    object
 12  MatchingFunds        725 non-null    object
 13  EstAvailFunds        713 non-null    object
 14  EstAwards            725 non-null    object
 15  EstAmounts           725 non-null    object
 16  FundingMethod        719 non-null    object
 17  OpenDate             720 non-null    object
 18  ApplicationDeadline  715 non-null    object
 19  ExpAwardDate         713 non-null    object
 20  GrantURL             725 non-null    object
dtypes: object(21)
memory usage: 119.1+ KB
None
```

CONVERTING NUMERIC VARIABLES

	FULL NAME	PREVIOUS DTYPE	DESCRIPTION
EstAwards	'Estimated Awards'	String	It's the closest planet to the Sun and the smallest one
EstAmounts	'Estimated Amounts'	String	Despite being red, Mars is actually a cold place
EstAvailFunds	'Estimated Available Funds'	String	It has a nice name and is the second planet from the Sun

CONVERTING NUMERIC VARIABLES

"EstAvailFunds" is formatted as a dollar amount in string form. Hence, we only need to remove the dollar sign if necessary and convert the string into an integer.

All entries in the other variables are in a consistent format:
"Between x and y" or *"Exactly z"*
Where x,y,z are numbers indicating a dollar amount

Thus our method of choice goes as follows:

1. Create an empty list for both max/min values for "EstAmounts" and "EstAwards"
2. Utilize a for-loop to iterate across all values of each column
3. Check to see if the first character in the each begins with a "B" to indicate a (1) "Between x and y" or an "E" to indicate an (2) "Exactly z"
4. If our string falls under case (1), we split the string at the "a" in "Between x and y", strip any non numeric characters in our two strings, and convert both x and y into integers
5. Append our min list with our integer x and max with with our integer y
6. For case (2) we strip all non numeric characters, convert z into an integer data type, and append both our max and min list with the integer z

```
[12]: awards = grants['EstAwards']
maxaward = []
minaward = []
for i in (range(len(awards))):
    if awards[i][0] == 'E':
        maxaward.append(int(''.join(filter(str.isdigit, awards[i]))))
        minaward.append(int(''.join(filter(str.isdigit, awards[i]))))
    elif awards[i][0] == 'B':
        maxaward.append(int(''.join(filter(str.isdigit, awards[i].rpartition('a')[2]))))
        minaward.append(int(''.join(filter(str.isdigit, awards[i].rpartition('a')[0]))))
    else:
        maxaward.append(float('nan'))
        minaward.append(float('nan'))


amounts = grants['EstAmounts']
maxamnt = []
minamnt = []
for i in (range(len(amounts))):
    if amounts[i][0] == 'E':
        maxamnt.append(int(''.join(filter(str.isdigit, amounts[i]))))
        minamnt.append(int(''.join(filter(str.isdigit, amounts[i]))))
    elif amounts[i][0] == 'B':
        maxamnt.append(int(''.join(filter(str.isdigit, amounts[i].rpartition('a')[2]))))
        minamnt.append(int(''.join(filter(str.isdigit, amounts[i].rpartition('a')[0]))))
    else:
        maxamnt.append(float('nan'))
        minamnt.append(float('nan'))
```

```
[13]: grants['MaxAwards'] = maxaward
grants['MinAwards'] = minaward
grants = grants.drop('EstAwards', axis = 1)

grants['MaxAmounts'] = maxamnt
grants['MinAmounts'] = minamnt
grants = grants.drop('EstAmounts', axis = 1)
```

```
[14]: availfunds = []
for i in (range(len(grants['EstAvailFunds']))):
    if type(grants['EstAvailFunds'][i]) != str:
        availfunds.append(float('nan'))
    else:
        availfunds.append(int(''.join(filter(str.isdigit, grants['EstAvailFunds'][i]))))
```

```
[15]: grants['EstAvailFunds'] = availfunds
grants.head()
```

Now by setting these lists as new columns we are left with 5 new numeric variables:

- MaxAmounts
- MinAmounts
- MaxAwards
- MinAwards
- EstAvailFunds

And we can delete or replace our old variables

```
[16]: print(grants.info()) # Our new columns are left with mostly missing values as a majority of entries were undeclared
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 725 entries, 0 to 724
Data columns (total 23 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   PortalID             725 non-null    object
 1   Status               725 non-null    object
 2   LastUpdated          725 non-null    object
 3   AgencyDept           725 non-null    object
 4   Title                725 non-null    object
 5   Type                 725 non-null    object
 6   LOI                  724 non-null    object
 7   Categories           725 non-null    object
 8   Purpose              724 non-null    object
 9   Description          725 non-null    object
 10  ApplicantType        721 non-null    object
 11  FundingSource        721 non-null    object
 12  MatchingFunds        725 non-null    object
 13  EstAvailFunds        713 non-null    float64
 14  FundingMethod        719 non-null    object
 15  OpenDate             720 non-null    object
 16  ApplicationDeadline  715 non-null    object
 17  ExpAwardDate         713 non-null    object
 18  GrantURL             725 non-null    object
 19  MaxAwards            148 non-null    float64
 20  MinAwards            148 non-null    float64
 21  MaxAmounts           231 non-null    float64
 22  MinAmounts           231 non-null    float64
dtypes: float64(5), object(18)
memory usage: 130.4+ KB
None
```

Summary Statistics for Numeric Variables

In [18]:
```
grants.describe()
```

Out[18]:

	EstAvailFunds	MaxAwards	MinAwards	MaxAmounts	MinAmounts
count	7.130000e+02	148.000000	148.000000	2.310000e+02	2.310000e+02
mean	6.380992e+07	3834.182432	693.114865	4.843356e+07	3.098185e+05
std	3.500748e+08	41384.329800	8218.712299	4.652575e+08	1.404581e+06
min	1.000000e+00	0.000000	0.000000	1.380000e+02	0.000000e+00
25%	1.170000e+06	2.000000	1.000000	1.000000e+05	0.000000e+00
50%	5.000000e+06	7.000000	2.000000	3.500000e+05	5.000000e+03
75%	2.000000e+07	20.000000	12.250000	1.500000e+06	5.000000e+04
max	5.000000e+09	500000.000000	100000.000000	5.000000e+09	1.500000e+07

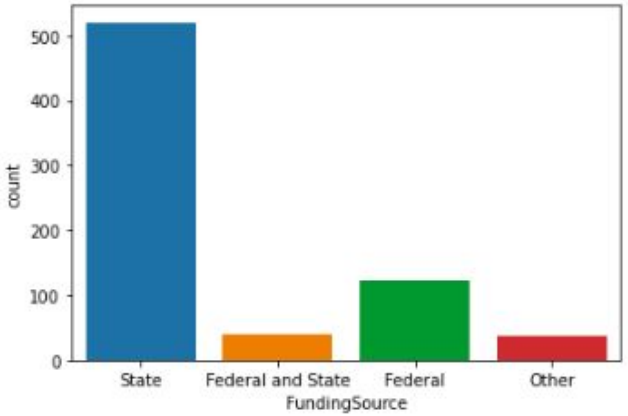
- Extreme variation in amounts and awards → large outliers

Variable Exploration

Funding Source vs Max Awards

```
In [27]: sns.countplot(x = 'FundingSource', data = grants) #to be expected as we are dealing with CA

Out[27]: <AxesSubplot:xlabel='FundingSource', ylabel='count'>
```



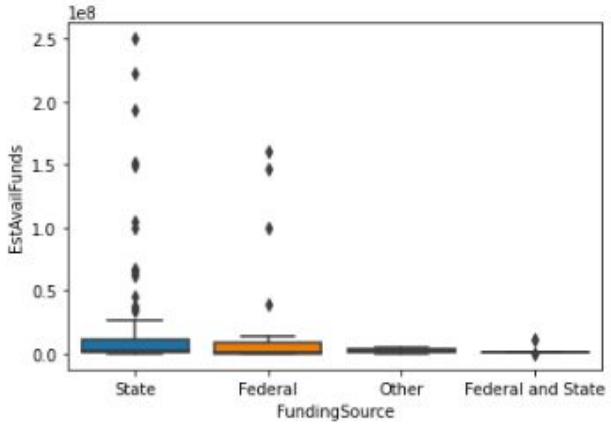
Looking more closely into funding source we find

- Majority of grants are sourced by the state

Funding Source vs Max Awards

In [26]:
```
# Potential relationship: Funding Source and Maximum Awards?
sns.boxplot(x = 'FundingSource', y = 'EstAvailFunds', data = grants2) #bulk of outliers are coming from state grants
```

Out[26]:
```
<AxesSubplot:xlabel='FundingSource', ylabel='EstAvailFunds'>
```

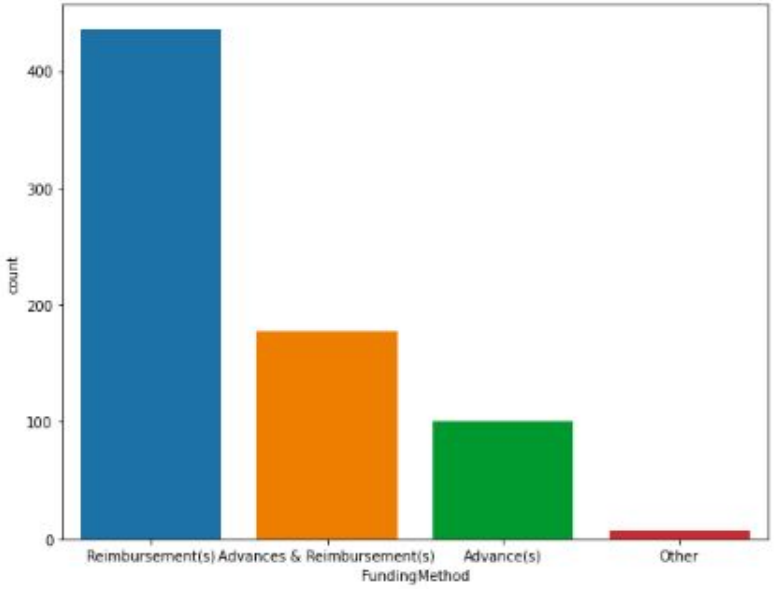


- Subsetted data to remove extreme outliers from MaxAwards
- Bulk of outliers coming from state grants
- Interesting point: state funding of grants > federal funding (for California)

Funding Method

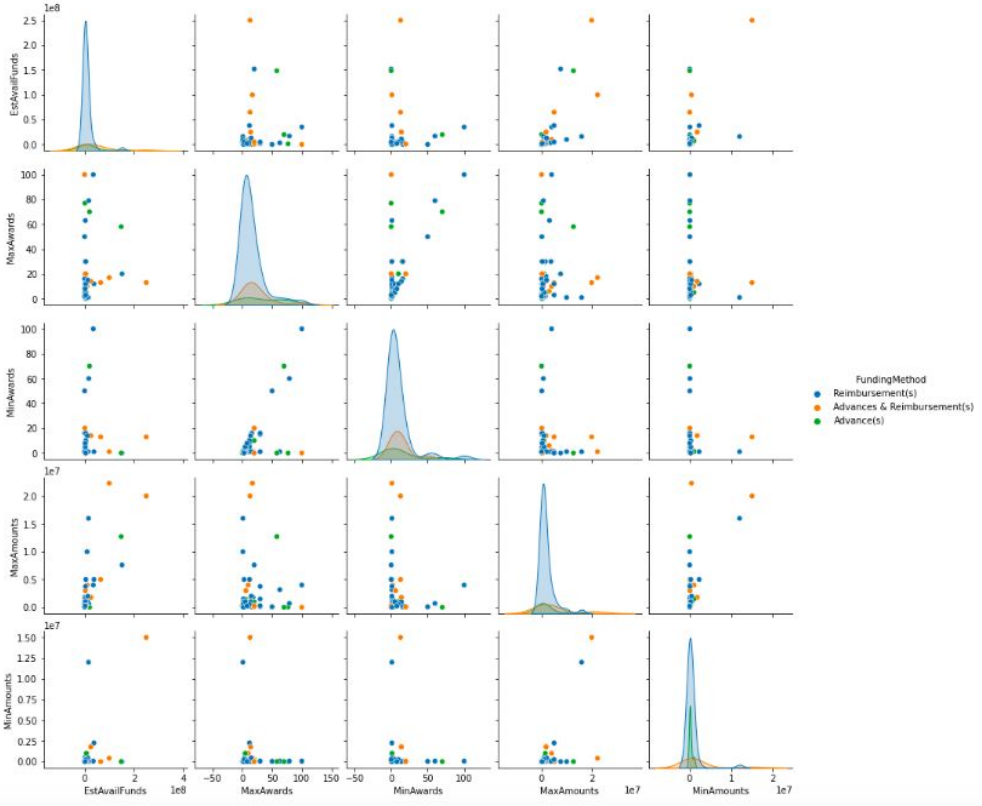
In [28]:
```
# Another potentially interesting variable to consider: Funding Method
fig, ax = plt.subplots()
fig.set_size_inches(9,7)
sns.countplot(x = 'FundingMethod', data = grants, ax = ax)
```

Out[28]:
```
<AxesSubplot:xlabel='FundingMethod', ylabel='count'>
```



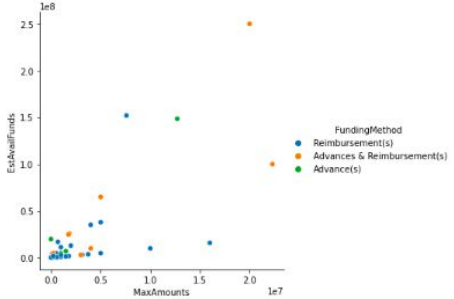
Numeric Scatterplots

Out[42]: <seaborn.axisgrid.PairGrid at 0x2610a6a2df0>



In [41]:
```
# Most interesting scatter: Maximum Amount vs Estimated Available Funds?
sns.relplot(x = 'MaxAmounts', y = 'EstAvailFunds', hue = 'FundingMethod', data = grants4)
sns.displot(data = grants4, x = 'MaxAmounts', y = 'EstAvailFunds')
```

Out[41]: <seaborn.axisgrid.FacetGrid at 0x2610a5d1d00>



Next Steps

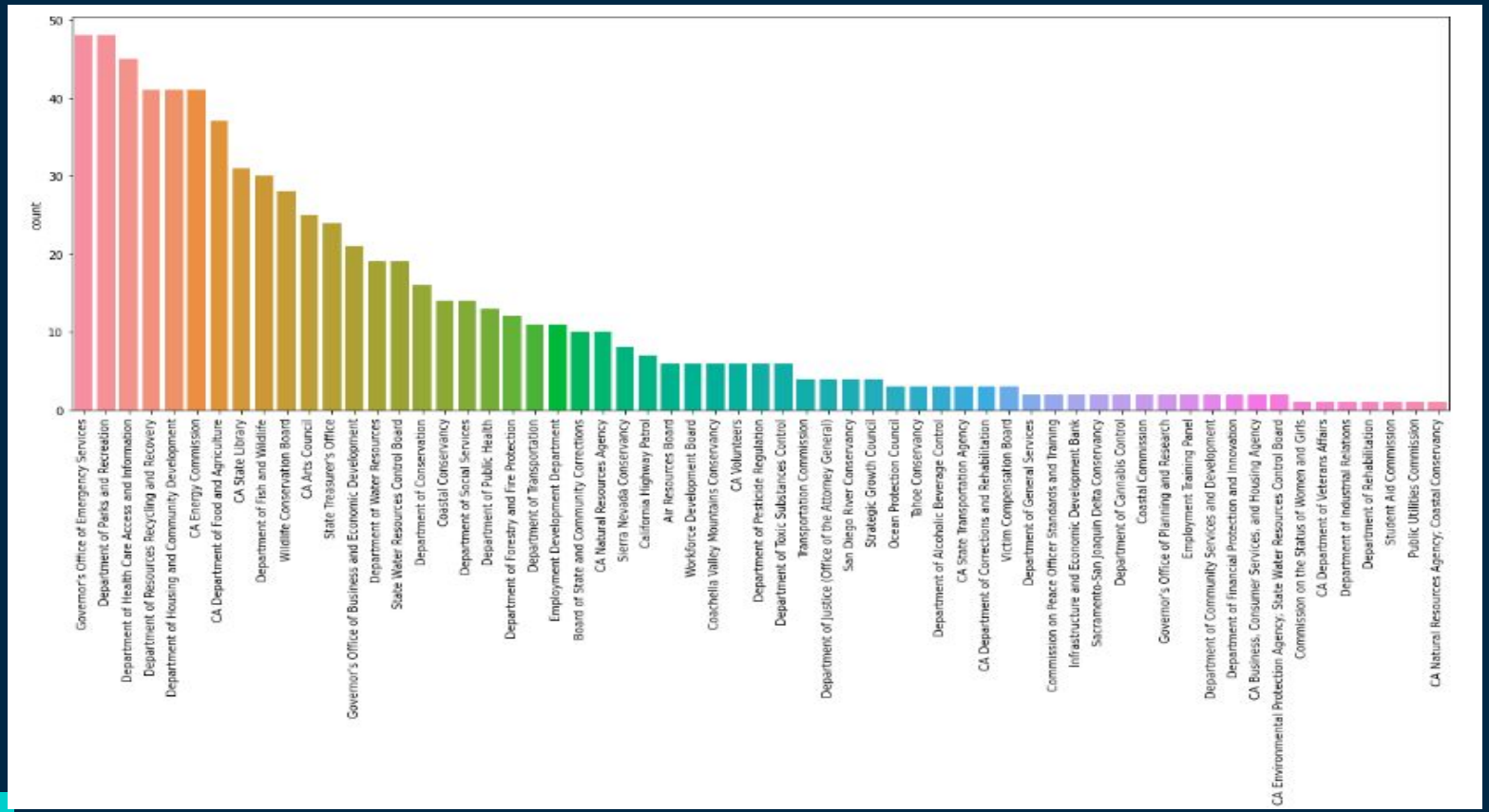
- Apply transformation
- LInear regression/analysis
- Multivariate regression

Grant Dates of Availability

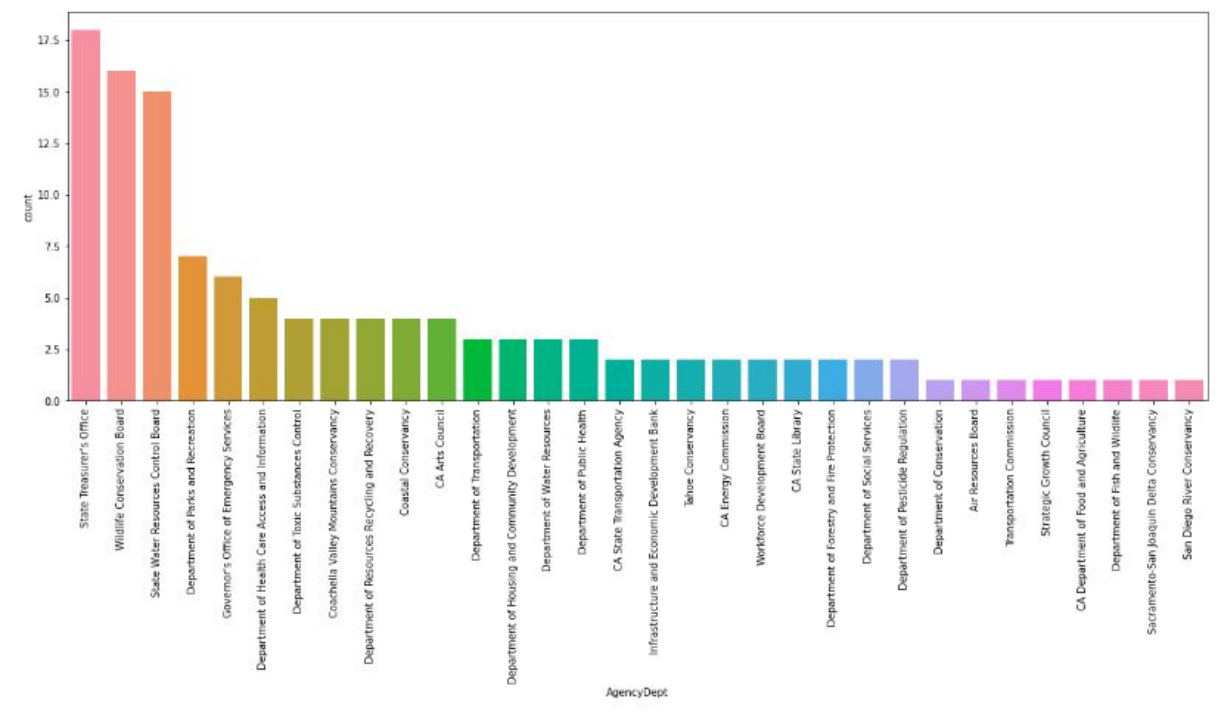
- Looking at the 'ApplicationDeadline' column for the dataset we can see how some grants are listed as 'Ongoing' while some are long past overdue.
- Since this dataset is constantly being updated → write a generalized function that will return a subset of the data with only ongoing grants
- Can be used for future iterations of this dataset

```
deadline = grants.ApplicationDeadline
ongoing = []
for i in deadline:
    if type(i) == float:
        ongoing.append(0)
    elif i[0] == 'O':
        ongoing.append(1)
    elif i[0] == '2':
        temp = pd.to_datetime(i, format="%Y-%m-%d %H:%M:%S")
        today = pd.datetime.now()
        if temp < today:
            ongoing.append(0)
        else:
            ongoing.append(1)
grants['IsOngoing'] = ongoing
```

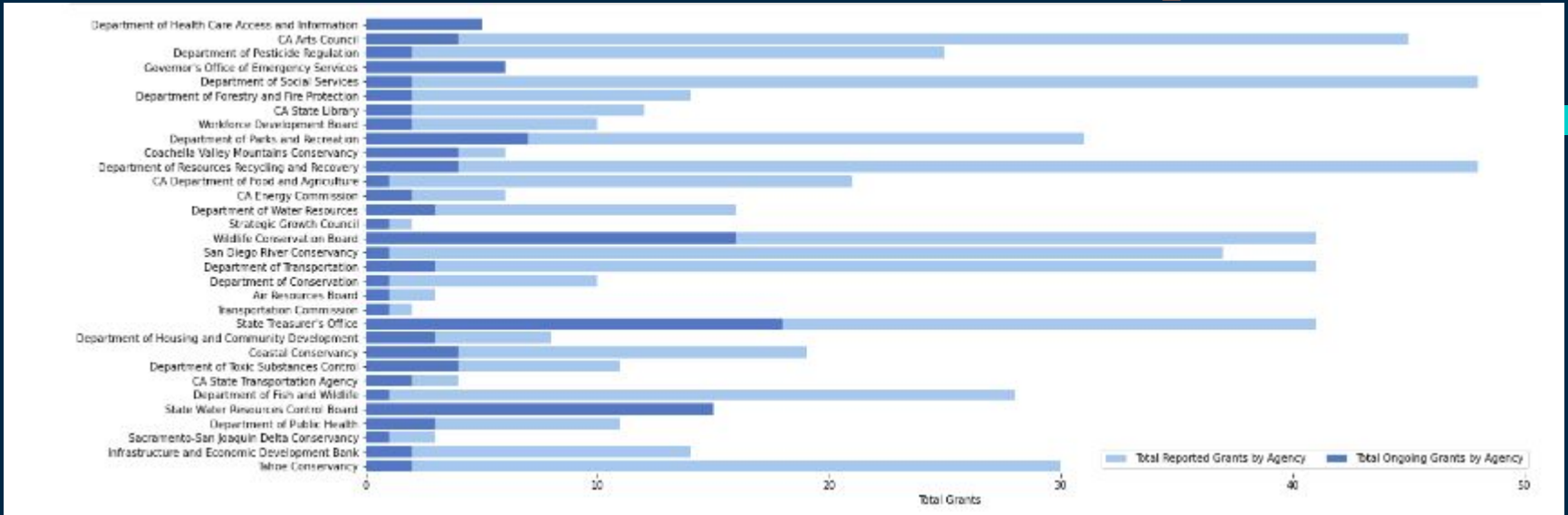
Agencies – Which are Reporting?



Ongoing Grants



- We can see a large dip in the amount of different agencies reporting
- Big difference in the rankings of reporting agencies



- When looking at the visualized difference in agency reporting for ongoing grants vs closed grants we see how significant this difference is
- This visualization doesnt even show the agencies who have no ongoing grants

Grant Categories

In [36]:

```
# dictionary to tally each different category
cat = {}
for i in grants_ongoing.Categories:
    for j in i.split('; '):
        if j not in cat:
            cat[j] = 1
        else:
            cat[j] += 1

categories_data = pd.DataFrame({'category': cat.keys(), 'count': cat.values()})
categories_data
```

Out[36]:

	category	count
0	Health & Human Services	16
1	Disadvantaged Communities	40
2	Libraries and Arts	5
3	Education	14
4	Employment, Labor & Training	9
5	Agriculture	10
6	Environment & Water	72
7	Food & Nutrition	3
8	Housing, Community and Economic Development	17
9	Parks & Recreation	21
10	Science, Technology, and Research & Development	9
11	Law, Justice, and Legal Services	6
12	Energy	11
13	Consumer Protection	3
14	Disaster Prevention & Relief	9
15	Transportation	11

- Every grant has one or multiple categories
- Categories of grants listed in the 'Category' column
- Want to observe categories more closely

Idea:

- Create a function to tally all unique categories found within the column

Implementing Binary Columns for Categories

- Will be easier to analyze grant category data with the implementation of binary columns for each unique category
- 1 → grant falls under that category; 0 → grant does not

Idea:

- Write a function that will iterate across all grants and create our binary columns

Creating Binary columns for Unique Categories

```
grants.Categories[23] # need to account for the space after the semicolon -> split at "; " ?
```

```
'Disadvantaged Communities; Health & Human Services'
```

```
# first attempt: error due to extra space in some entries --> strip whitespace
category = pd.read_csv('category_table.csv')
cat = {}
for i in list(category.category):
    cat[i] = []
for i in grants_ongoing.Categories:
    temp = []
    for j in i.split('; '):
        temp.append(j)
    for key in cat:
        if key not in temp:
            cat[key].append(0)
        else:
            cat[key].append(1)
cat_df = pd.DataFrame(cat)
frames = [grants_ongoing, cat_df]
grants_ongoingBinary = pd.concat(frames, axis = 1)
```

Output

```
# viewing index numbers for iloc
grants_ongoingBinary.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 125 entries, 0 to 124
Data columns (total 40 columns):
 #   Column                                          Non-Null Count  Dtype
---  ------                                          --------------  -----
 0   PortalID                                        125 non-null    int64
 1   Status                                          125 non-null    object
 2   LastUpdated                                     125 non-null    object
 3   AgencyDept                                      125 non-null    object
 4   Title                                           125 non-null    object
 5   Type                                            125 non-null    object
 6   LOI                                             125 non-null    object
 7   Categories                                      125 non-null    object
 8   Purpose                                         125 non-null    object
 9   Description                                     125 non-null    object
 10  ApplicantType                                   125 non-null    object
 11  FundingSource                                   125 non-null    object
 12  MatchingFunds                                   125 non-null    object
 13  EstAvailFunds                                   121 non-null    float64
 14  FundingMethod                                   125 non-null    object
 15  OpenDate                                        125 non-null    object
 16  ApplicationDeadline                             125 non-null    object
 17  ExpAwardDate                                    122 non-null    object
 18  GrantURL                                        125 non-null    object
 19  MaxAwards                                       17 non-null     float64
 20  MinAwards                                       17 non-null     float64
 21  MaxAmounts                                      28 non-null     float64
 22  MinAmounts                                      28 non-null     float64
 23  IsOngoing                                       125 non-null    int64
 24  Health & Human Services                         125 non-null    int64
 25  Disadvantaged Communities                       125 non-null    int64
 26  Libraries and Arts                              125 non-null    int64
 27  Education                                       125 non-null    int64
 28  Employment, Labor & Training                    125 non-null    int64
 29  Agriculture                                     125 non-null    int64
 30  Environment & Water                             125 non-null    int64
 31  Food & Nutrition                                125 non-null    int64
 32  Housing, Community and Economic Development     125 non-null    int64
 33  Parks & Recreation                              125 non-null    int64
 34  Science, Technology, and Research & Development 125 non-null    int64
 35  Law, Justice, and Legal Services                125 non-null    int64
 36  Energy                                          125 non-null    int64
 37  Consumer Protection                             125 non-null    int64
 38  Disaster Prevention & Relief                    125 non-null    int64
 39  Transportation                                  125 non-null    int64
dtypes: float64(5), int64(18), object(17)
memory usage: 39.2+ KB
```

```
# confirming for correctness
index = [7] + list(range(24,39))
grants_ongoingBinary.iloc[:,index]
```

	Categories	Health & Human Services	Disadvantaged Communities	Libraries and Arts	Education	Employment, Labor & Training	Agriculture	Environment & Water	Food & Nutrition	Housing, Community and Economic Development	Parks & Recreation	Science, Technology, and Research & Development	Law, Justice, and Legal Services
0	Health & Human Services	1	0	0	0	0	0	0	0	0	0	0	0
1	Disadvantaged Communities; Libraries and Arts	0	1	1	0	0	0	0	0	0	0	0	0
2	Disadvantaged Communities; Education; Employme...	0	1	1	1	1	0	0	0	0	0	0	0
3	Agriculture; Disadvantaged Communities; Educat...	0	1	0	1	1	1	1	1	1	1	1	0
4	Agriculture; Disadvantaged Communities; Educat...	0	1	0	1	1	1	1	1	1	1	1	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
120	Education	0	0	0	1	0	0	0	0	0	0	0	0
121	Disadvantaged Communities; Disaster Prevention...	0	1	0	0	1	0	1	0	1	0	1	0
122	Education	0	0	0	1	0	0	0	0	0	0	0	0
123	Education	0	0	0	1	0	0	0	0	0	0	0	0
124	Environment & Water	0	0	0	0	0	0	1	0	0	0	0	0

125 rows × 16 columns

Exploring Grants' Categories Based on Descriptions

The 'Description' variable potentially has a lot of potentially useful information. We will see the relationship between grants' descriptions for different sets of categories.

Idea:

- Write a function that analyzes grants' descriptions based on their categories
- Takes a list of indices (of binary category columns) as well as a dataset
- Finds all grants with that specific set of categories and perform a keyword analysis
- Explore what this indicated about the relationship of these categories

Code and Test Cases

[12]:
```
import spacy
nlp = spacy.load('en_core_web_sm')
# Idea: function that takes in categories (through a list in indices) and a dataset -> takes the descriptions of the entries with those categories -> returns keywords
# can be polished for later project and used with the larger dataset

# 3 categories: Disadvantaged communities; Education; Housing, Community and Economic Development
CatIn = [25,27,32] # index numbers for columns

def keywords(categories, data):
    # setting up password (to select entries with all desired categories)
    text = ""
    password = []
    for i in categories:
        password.append(1)

    # checking password and obtaining descriptions
    for j in range(len(data)):
        if list(data.iloc[j,categories]) == password:
            text = text + ' ' + data.iloc[j,9]

    # finding keywords
    out = nlp(text)
    print(out.ents)

# test case 1
keywords(CatIn, grants_ongoingBinary)
```

```
(DPR, Alliance Grants Program, $1.5 million, Alliance, Integrated Pest Management, IPM, IPM, IPM, Alliance Grant, Alliance Grants Program, 2023, DPR, DPR, the Pest Manageme
nt Advisory Committee, PMAC, the Proposal Application, PMAC, PMAC, DPR, DPR, $1.5 million, 50,000, $1.5 million, California, California, the PMAC Charter, DPR, Agronomy, Ai
r Quality, Automation, Community Health, Cover Crops, Cropping System, Crops, Ecology, Ecosystem, Fauna, Flora, Fumigant, Fungi, Fungicide, Herbicide,, Horticulture, Integr
ated Pest Management, Irrigation, Lakes, Land Management, Mating Disruption, Miticide, Natural Enemies, Oceans, Pathogens, Personal Protective Equipment, Pest, Pest Managem
ent, Pesticide, Pollinator, Pollution, Reduced-Risk, Rivers, Rodenticide, Soil Health, Streams, Sustainable, Training, Urban Pest Management, Vegetables, Vertebrate Pests,
Virus, Watershed, Worker Health and Safety DPR's, Research Grants Program, IPM, DPR, Research Grants Program, a Proposal Application, DPR, the Pest Management Advisory Comm
ittee, PMAC, the Proposal Application, PMAC, PMAC, DPR, DPR, $3.15 million, $50,000 to $3.15 million, California, California, the PMAC Charter, DPR, DPR, County Agricultura
l Commissioner, Keywords, Agronomy, Air Quality, Automation, Community Health, Cover Crops, Cropping System, Crops, Ecology, Ecosystem, Fauna, Flora, Fumigant, Fungi, Fungi
cide, Herbicide,, Horticulture, Integrated Pest Management, Irrigation, Lakes, Land Management, Mating Disruption, Miticide, Natural Enemies, Oceans, Pathogens, Personal Pr
otective Equipment, Pest, Pest Management, Plant Protection, Soil Health, Streams, Sustainable, Training, Urban Pest Management, Vegetables, Vertebrate Pests, Virus, Waters
hed, Worker Health and Safety, The Community and Economic Enhancement Grant Program, Delta, Delta, today, Disadvantaged/Severely Disadvantaged Community, California Environ
mental Quality Act, Delta Plan, 15 years, Conservancy, Conservancy, Conservancy, Board)
```

Code and Test Cases

```
[13]: # test case 2

CatIn = [25,27] # 2 categories
keywords(CatIn, grants_ongoingBinary)
```

```
(approximately 11, 12-month, California, CAC, AO, DPR, Alliance Grants Program, $1.5 million, Alliance, Integrated Pest Management, IPM, IPM, IPM, Alliance Grant, Alliance
Grants Program, 2023, DPR, DPR, the Pest Management Advisory Committee, PMAC, the Proposal Application, PMAC, PMAC, DPR, DPR, $1.5 million, 50,000, $1.5 million, Californi
a, California, the PMAC Charter, DPR, Agronomy, Air Quality, Automation, Community Health, Cover Crops, Cropping System, Crops, Ecology, Ecosystem, Fauna, Flora, Fumigant,
Fungi, Fungicide, Herbicide,, Horticulture, Integrated Pest Management, Irrigation, Lakes, Land Management, Mating Disruption, Miticide, Natural Enemies, Oceans, Pathogens,
Personal Protective Equipment, Pest, Pest Management, Pesticide, Pollinator, Pollution, Reduced-Risk, Rivers, Rodenticide, Soil Health, Streams, Sustainable, Training, Urba
n Pest Management, Vegetables, Vertebrate Pests, Virus, Watershed, Worker Health and Safety DPR's, Research Grants Program, IPM, DPR, Research Grants Program, a Proposal Ap
plication, DPR, the Pest Management Advisory Committee, PMAC, the Proposal Application, PMAC, PMAC, DPR, DPR, $3.15 million, $50,000 to $3.15 million, California, Californi
a, the PMAC Charter, DPR, DPR, County Agricultural Commissioner, Keywords, Agronomy, Air Quality, Automation, Community Health, Cover Crops, Cropping System, Crops, Ecolog
y, Ecosystem, Fauna, Flora, Fumigant, Fungi, Fungicide, Herbicide,, Horticulture, Integrated Pest Management, Irrigation, Lakes, Land Management, Mating Disruption, Miticid
e, Natural Enemies, Oceans, Pathogens, Personal Protective Equipment, Pest, Pest Management, Plant Protection, Soil Health, Streams, Sustainable, Training, Urban Pest Manag
ement, Vegetables, Vertebrate Pests, Virus, Watershed, Worker Health and Safety, The Advanced Practice Healthcare Scholarship Program, up to $25,000, one year, 12-month, Ca
lifornia, The Allied Healthcare Scholarship Program, up to $25,000, one year, 12-month, California, The Community and Economic Enhancement Grant Program, Delta, Delta, toda
y, Disadvantaged/Severely Disadvantaged Community, California Environmental Quality Act, Delta Plan, 15 years, Conservancy, Conservancy, Conservancy, Board)
```

- Above gives us the keywords for the categories: Disadvantaged Communities and Education
- The sheer volume of keywords may be too abstract to draw any meaningful conclusions
- We also don't have a general structure of how these words are comparatively significant

Yake Library

Using the Yake library we can perform a similar keyword analysis with the added benefits of our own custom customizations. These include:

- Length of phrases
- Controlling of repetitivity
- Controlling amount of keywords outputted

Additionally, for every keyword we also get a number to signify its significance level.

Test Case: Disadvantaged communities; Education; Housing, Community and Economic Development

```
CatIn = [25,27,32]
# test case 3
keywords_yake(CatIn, grants_ongoingBinary)
```

```
('Pest Management', 0.0032835720406815924)
('Integrated Pest Management', 0.0038654970575418743)
('Alliance Grants Program', 0.004927899043113235)
('Pest Management Advisory', 0.00630150636171292)
('Urban Pest Management', 0.006339878789703293)
('DPR Alliance Grants', 0.006644159717626999)
('Alliance Grants application', 0.011384082526051062)
('Alliance Grants', 0.011646173030221838)
('Research Grants Program', 0.013239053343462238)
('Alliance Grant projects', 0.013483373219107858)
('Pest', 0.013497821909898823)
('Grants Program', 0.01357881920035414)
('DPR', 0.013809928479134497)
('PMAC', 0.015085992550257285)
('Management', 0.01611027984066794)
('DPR Research Grants', 0.017552094310763468)
('Management Advisory Committee', 0.017816846108178405)
('projects', 0.018131142400430976)
('Integrated Pest', 0.018380139258986506)
('affordable Integrated Pest', 0.018424696840988325)
```

Yake Library

With this we could also see which categories are the most correlated in terms of descriptions.

We would need to implement a function that sums the total keyword significance of all pairs of categories and return the maximum.

The output indicated that ‘Housing, Community and Economic Development’ and ‘Law, Justice, and Legal Services’ have the most similar descriptions.

(Results on the Right)

```
[39]: keywords_yake([32,35], grants_ongoingBinary)

('Penal Code', 0.004811108706349856)
('Program Components', 0.013889889267394498)
('domestic violence', 0.047515392189801456)
('including emergency shelter', 0.049569384002041854)
('domestic violence services', 0.060479130124080106)
('Code', 0.06263145762688499)
('Components', 0.06263145762688499)
('emergency shelter', 0.06355622323368645)
('Subrecipients must provide', 0.06498425530544041)
('Program provides local', 0.06630686322886482)
('supportive services', 0.06753840813728693)
('Penal', 0.07644836874616959)
('Subrecipients', 0.07644836874616959)
('provide access', 0.07657948748239458)
('non-English speaking individuals', 0.0860313059940345)
('services', 0.08676560611836971)
('Program provides', 0.09331874890674917)
('domestic violence providers', 0.10464876506821895)
('existing domestic violence', 0.10464876506821896)
('domestic', 0.10911176996890641)
```