Rishab Kedia

Kevin Liu

Jeffrey Chen

Jaydon Leo Krooss

Data Science Project 1 Write-Up

For the project, our group worked with the School System Finances data set. Before we could read our data into our python file, we had to convert it to a csv type file. Once we had our data loaded, we looked at a summary of our data and the documentation, and realized we only wanted to use a few columns for the data for our analysis, so we stored a subset of our data that included the state of the school system, year the data was recorded, as well as the total revenue and its division among federal, state, and local sources. However, we also realized that we may want to later compare this data to the school's spending, which was a completely different aspect of its finances. For this reason, we separately stored another subset of the data with information about the total spending for a school program and how it was divided among school instruction, support services, and other programs.

To clean our data, we used the dropna( ) function to get rid of empty entries. However, when we checked the data set initially with the missingno library, there did not seem to be very many missing entries at all, so our group believed that dropping the empty entries may not have affected our data very much, if at all. Here are a few examples of data that we cleaned:

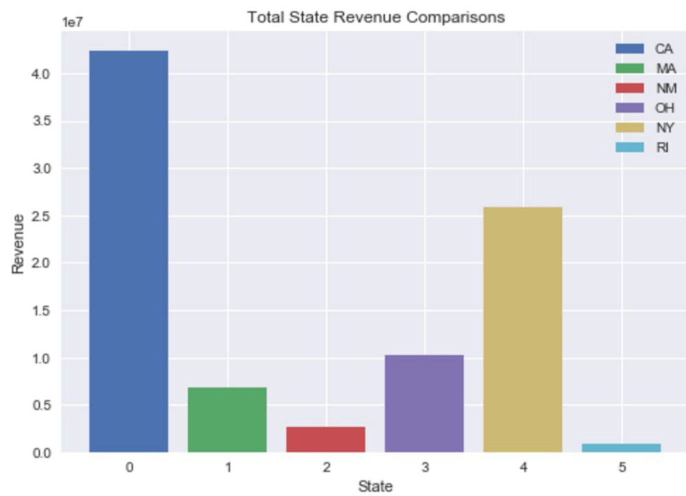
```
clean_useful.describe()
```

	STATE	YRDATA	TOTALREV	TFEDREV	TSTREV
count	14376.000000	14376.0	1.437600e+04	1.437600e+04	1.437600e+04
mean	26.801336	15.0	4.546951e+04	3.708038e+03	2.105122e+04
std	13.894331	0.0	2.590623e+05	2.021328e+04	1.125639e+05
min	1.000000	15.0	0.000000e+00	0.000000e+00	0.000000e+00
25%	15.000000	15.0	5.241500e+03	2.920000e+02	2.132500e+03
50%	27.000000	15.0	1.402300e+04	8.470000e+02	6.516000e+03
75%	38.000000	15.0	3.791375e+04	2.433500e+03	1.667925e+04
max	51.000000	15.0	2.543738e+07	1.307783e+06	9.837509e+06

```
clean_spending.head()
```

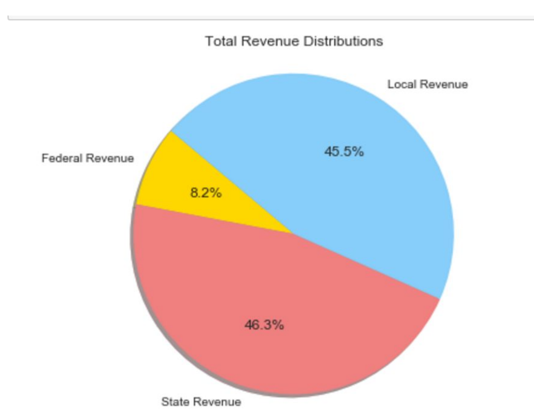
	TCURELSC	TCURINST	TCURSSVC	TCUROTH
0	72872	44085	23217	5570
1	269928	155668	99682	14578
2	9957	5249	3835	873
3	24232	14887	7494	1851
4	29133	16019	10822	2292

After we had loaded and cleaned our data, we began making different visual representations of the data to extract meaningful information from it. The first graph (shown below) we made was a bar graph that compared the revenue of schools from six different states: California, Massachusetts, New Mexico, Ohio, New York, and Rhode Island.



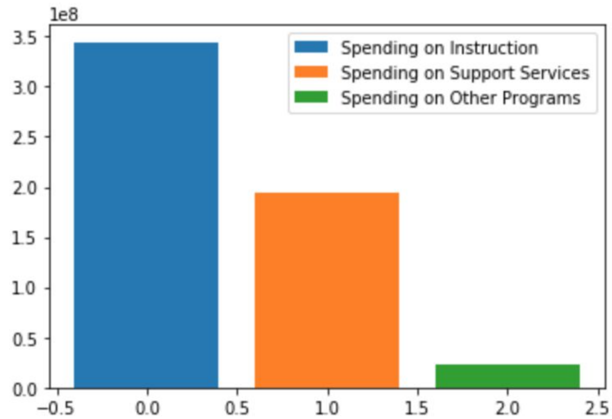
From this graph, we can see that out of the states that were compared, California schools receive the most state revenue, with New York coming as a distant second. The plot also indicates that Rhode Island schools receive the least state funding out of the schools listed, though it is definitely biased as Rhode Island is a much smaller state than the other states.

The next visualization we created was a pie chart that depicts how much funding schools get from federal, state, and local sources relative to each other, as seen below:



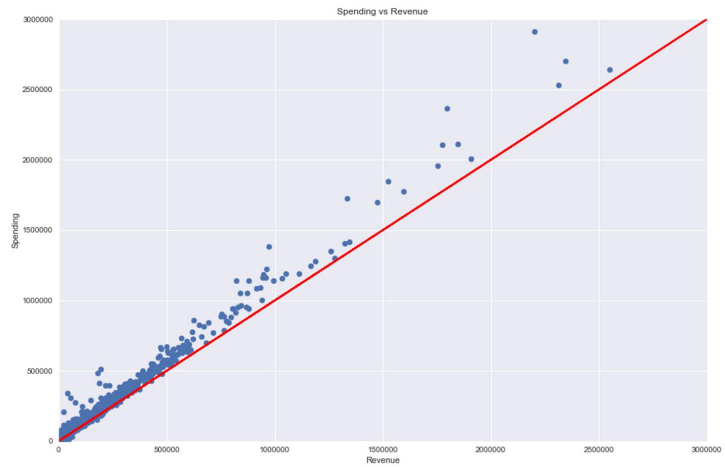
From this chart, we realized that elementary and secondary schools do not receive much federal funding, and rely more heavily on local and state funding. Local and state funding contribute almost equally to elementary and secondary schools.

We also created another visualization that focused on how schools spend their money, which is shown below:



From this figure, we learned that elementary and secondary schools seem to spend the most money on instruction relative to their support services and other school programs. In fact, it appears that spending on various other school programs is far less than on instruction.

Additionally, another aspect of school finances our group was interested in was how schools spent money relative to how much funding they received. For this reason we made a linear regression model that depicted Spending vs. Revenue, with revenue acting as our explanatory variable because we expected that a school's spending depended on how much funding they received. This linear regression model is shown below:



Since the data seemed linearly distributed, a linear regression model seemed very appropriate. Our model could be used to predict how much a school should be spending given its amount of funding. However, our prediction line seemed to be a little low, rather than through our data, and we think this may be due to a couple of outliers that did not fit in the chart when we zoomed in on the rest of the data. We could make the regression better if we were able to remove the outliers.

Finally, we also tried to use logistic regression on our data set. We did not really have categorical data that would lend itself well to logistic regression, but we thought about a couple of different possible models, all of which did not perform well under a logistical model since spending and revenue are extremely linearly correlated. We tried a few ways of implementing Logistic Regression, but in the end decided upon the shorter approach of importing LogisticRegression from Scikit-Learn, and used their function rather than writing one up ourselves. Here is a model we ran, and it turns out we were correct that there wasn't any logistical correlation between the two variables as the correlation was extremely low.

```python
from sklearn.linear_model import LogisticRegression
logistic = LogisticRegression()
logistic.fit(rev,spend)
```

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='ovr',
          penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
          verbose=0)
```

```python
lr = LogisticRegression()
X = rev
y = spend
lr.fit(X, y)
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, penalty='l2', random_state=None, tol=0.0001)
prob = lr.predict_proba(X[0])
```

```python
prob[0][0]
```

```
2.1723278868657562e-37
```

Project-1-School-System-Finances

October 3, 2017

In [44]:
```
import pandas as pd
import statsmodels.api as sm
from sklearn.feature_selection import RFECV
from sklearn.linear_model import LinearRegression
from scipy.io import loadmat as loadmat
import matplotlib
import numpy as np
import seaborn as sns
import matplotlib.cm as cm
import matplotlib.mlab as mlab
import matplotlib.pyplot as plt
import math

# WE CHOSE THE SCHOOL SYSTEM FINANCES DATA SET
```

In [45]:
```
%matplotlib inline
```

In [46]:
```
df = pd.read_csv('elsec15-Table 1.csv', low_memory=False)
```

In [47]:
```
df
```

Out[47]:

	STATE	IDCENSUS	NAME	CONUM	CSA	\
0	1	1500100100000	AUTAUGA COUNTY SCHOOL DISTRICT	1001	N	
1	1	1500200100000	BALDWIN COUNTY SCHOOL DISTRICT	1003	380	
2	1	1500300100000	BARBOUR COUNTY SCHOOL DISTRICT	1005	N	
3	1	1500300200000	EUFAULA CITY SCHOOL DISTRICT	1005	N	
4	1	1500400100000	BIBB COUNTY SCHOOL DISTRICT	1007	142	
5	1	1500500100000	BLOUNT COUNTY SCHOOL DISTRICT	1009	142	
6	1	1500500200000	ONEONTA CITY SCHOOL DISTRICT	1009	142	
7	1	1500600100000	BULLOCK COUNTY SCHOOL DISTRICT	1011	N	
8	1	1500700100000	BUTLER COUNTY SCHOOL DISTRICT	1013	N	
9	1	1500800200000	ANNISTON CITY SCHOOL DISTRICT	1015	N	
10	1	1500800300000	CALHOUN COUNTY SCHOOL DISTRICT	1015	N	
11	1	1500800400000	JACKSONVILLE CITY SCHOOL DISTRICT	1015	N	
12	1	1500800500000	PIEDMONT CITY SCHOOL DISTRICT	1015	N	
13	1	1500880100000	OXFORD CITY SCHOOL DISTRICT	1015	N	
14	1	1500900100000	CHAMBERS COUNTY SCHOOL DISTRICT	1017	194	
15	1	1500900200000	LANETT CITY SCHOOL DISTRICT	1017	194	
16	1	1501000100000	CHEROKEE COUNTY SCHOOL DISTRICT	1019	N	
17	1	1501100100000	CHILTON COUNTY SCHOOL DISTRICT	1021	142	
18	1	1501200100000	CHOCTAW COUNTY SCHOOL DISTRICT	1023	N	
19	1	1501300100000	CLARKE COUNTY SCHOOL DISTRICT	1025	N	
20	1	1501370100000	THOMASVILLE CITY SCHOOL DISTRICT	1025	N	
21	1	1501400100000	CLAY COUNTY SCHOOL DISTRICT	1027	N	

22	1	1501500100000	CLEBURNE COUNTY SCHOOL DISTRICT	1029	N
23	1	1501600100000	COFFEE COUNTY SCHOOL DISTRICT	1031	222
24	1	1501600200000	ELBA CITY SCHOOL DISTRICT	1031	222
25	1	1501600300000	ENTERPRISE CITY SCHOOL DISTRICT	1031	222
26	1	1501700100000	COLBERT COUNTY SCHOOL DISTRICT	1033	N
27	1	1501700200000	SHEFFIELD CITY SCHOOL DISTRICT	1033	N
28	1	1501700300000	TUSCUMBIA CITY SCHOOL DISTRICT	1033	N
29	1	1501750100000	MUSCLE SHOALS CITY SCHOOL DISTRICT	1033	N
...	...	...	...	...	...
14346	51	51500780200000	FREMONT CO SCHOOL DIST 6	56013	N
14347	51	51500800300000	GOSHEN CO SCH DIST 1	56015	N
14348	51	51500901100000	HOT SPRINGS CO SCH DIST 1	56017	N
14349	51	51501000200000	JOHNSON CO SCH DIST 1	56019	N
14350	51	51501100100000	LARAMIE CO SCH DIST 1	56021	N
14351	51	51501100700000	LARAMIE CO SCH DIST 2	56021	N
14352	51	51501200100000	LINCOLN CO SCH DIST 1	56023	N
14353	51	51501290100000	LINCOLN CO SCHOOL DIST 2	56023	N
14354	51	51501301200000	NATRONA CO SCH DIST 1	56025	N
14355	51	51501400100000	NIOBRARA CO SCH DIST	56027	N
14356	51	51501500100000	PARK CO SCH DISTRICT 1	56029	N
14357	51	51501500300000	PARK CO SCH DIST 6	56029	N
14358	51	51501500600000	PARK CO SCH DIST 16	56029	N
14359	51	51501600500000	PLATTE CO SCH DIST #1	56031	N
14360	51	51501670100000	PLATTE CO SCH DIST 2	56031	N
14361	51	51501700600000	SHERIDAN CO SCH DIST 2	56033	N
14362	51	51501701500000	SHERIDAN CO SCH DIST 3	56033	N
14363	51	51501701700000	SHERIDAN CO SCH DIST 1	56033	N
14364	51	51501800100000	SUBLETTE CO SCH DIST 1	56035	N
14365	51	51501800600000	BIG PINEY SCH DIST 9	56035	N
14366	51	51501900200000	SWEETWATER CO SCH DIST 2	56037	N
14367	51	51501900400000	SWEETWATER CO SCH DIST 1	56037	N
14368	51	51502000200000	TETON CO SCH DIST 1	56039	N
14369	51	51502100100000	UINTA CO SCH DIST 1	56041	N
14370	51	51502100200000	UINTA CO SCH DIST 4	56041	N
14371	51	51502100400000	UINTA CO SCH DIST 6	56041	N
14372	51	51502200300000	WASHAKIE SCH DIST 2	56043	N
14373	51	51502200400000	WASHAKIE CO SCH DIST 1	56043	N
14374	51	51502300100000	WESTON CO SCH DIST 1	56045	N
14375	51	51502300200000	WESTON CO SCHOOL DIST 7	56045	N

	CBSA	SCHLEV	NCESID	YRDATA	V33	...	V32	_19H	_21F	\
0	33860	3	0100240	15	9664	...	0	49431	16603	
1	19300	3	0100270	15	30596	...	0	337160	99087	
2	N	3	0100300	15	925	...	0	8024	0	
3	N	3	0101410	15	2829	...	0	0	0	
4	13820	3	0100360	15	3357	...	0	22155	0	
5	13820	3	0100420	15	8082	...	0	11123	0	
6	13820	3	0102550	15	1485	...	0	2598	0	
7	N	3	0100480	15	1519	...	0	4150	0	
8	N	3	0100510	15	3191	...	0	28128	31858	
9	11500	3	0100090	15	2141	...	0	2448	0	
10	11500	3	0100540	15	9232	...	0	1096	0	
11	11500	3	0101860	15	1544	...	0	9586	0	
12	11500	3	0102760	15	1223	...	0	2504	0	

13	11500	3	0102635	15	4205	...	0	28093	12498
14	46740	3	0100600	15	3739	...	0	18336	0
15	46740	3	0101980	15	942	...	0	596	0
16	N	3	0100630	15	4047	...	0	16780	3979
17	13820	3	0100660	15	7620	...	0	1683	0
18	N	3	0100690	15	1575	...	0	17860	0
19	N	3	0100720	15	2897	...	0	16039	402
20	N	3	0103300	15	1390	...	0	1857	0
21	N	3	0100750	15	2031	...	0	5799	0
22	N	3	0100780	15	2697	...	0	5287	300
23	21460	3	0100810	15	2028	...	0	9799	0
24	21460	3	0101260	15	658	...	0	1242	50
25	21460	3	0101320	15	6674	...	0	9635	0
26	22520	3	0100840	15	2703	...	0	420	0
27	22520	3	0103000	15	1118	...	0	3513	0
28	22520	3	0103420	15	1551	...	0	5911	3067
29	22520	3	0102520	15	2880	...	0	11557	150
...	...	...	...	...	...	...	...	...	...
14346	40180	3	5602830	15	359	...	0	0	0
14347	N	3	5602990	15	1713	...	0	0	0
14348	N	3	5603310	15	617	...	0	0	0
14349	N	3	5603770	15	1284	...	0	0	0
14350	16940	3	5601980	15	13761	...	0	0	0
14351	16940	3	5604120	15	987	...	0	0	0
14352	N	3	5604030	15	634	...	0	3850	0
14353	N	3	5604060	15	2681	...	0	0	0
14354	16220	3	5604510	15	13433	...	0	0	0
14355	N	3	5604230	15	978	...	0	0	0
14356	N	3	5605160	15	1761	...	0	0	0
14357	N	3	5602070	15	2105	...	0	0	0
14358	N	3	5604380	15	109	...	0	0	0
14359	N	3	5605090	15	1003	...	0	0	0
14360	N	3	5603180	15	228	...	0	0	0
14361	43260	3	5605695	15	3390	...	0	0	0
14362	43260	3	5605680	15	84	...	0	0	0
14363	43260	3	5605690	15	980	...	0	990	0
14364	N	3	5604860	15	1035	...	0	0	0
14365	N	3	5601260	15	627	...	48	0	0
14366	40540	3	5605762	15	2726	...	0	10500	0
14367	40540	3	5605302	15	5719	...	0	0	0
14368	27220	3	5605830	15	2691	...	0	0	0
14369	21740	3	5602760	15	2911	...	0	0	0
14370	21740	3	5604500	15	791	...	0	0	0
14371	21740	3	5604260	15	721	...	0	5000	0
14372	N	3	5605820	15	91	...	0	0	0
14373	N	3	5606240	15	1353	...	0	8610	0
14374	N	3	5604830	15	784	...	0	0	0
14375	N	3	5606090	15	264	...	0	0	0

	_31F	_41F	_61V	_66V	W01	W31	W61
0	2992	63042	0	0	2094	372	8617
1	13027	423220	0	0	5784	50441	71370
2	304	7720	0	0	0	0	646
3	0	0	0	0	0	2054	7478

4	1190	20965	0	0	1397	790	5400
5	980	10143	0	0	843	947	22610
6	248	2350	0	0	2190	814	1965
7	334	3816	0	0	324	0	1177
8	32248	27738	0	0	522	3845	3456
9	51	2397	0	0	1652	673	0
10	812	284	0	0	456	7148	17969
11	541	9045	0	0	115	8032	2754
12	256	2248	0	0	708	1434	771
13	12820	27772	0	0	1163	1053	9851
14	2032	16304	0	0	613	60	7405
15	189	407	0	0	0	0	1216
16	662	20097	0	0	339	3987	10539
17	1391	292	0	0	0	6074	11155
18	1241	16619	0	0	611	2335	1786
19	932	15508	0	0	105	528	1766
20	539	1318	0	0	283	177	4032
21	459	5340	0	0	71	577	958
22	689	4898	0	0	74	412	3709
23	364	9435	0	0	1059	432	8134
24	46	1246	0	0	0	180	493
25	722	8913	0	0	0	23247	9507
26	129	291	0	0	0	2847	5259
27	30	3483	0	0	206	567	5337
28	177	8801	0	0	634	5161	3872
29	0	11707	0	0	373	81	6836
...	...	...	...	...	...	...	...
14346	0	0	0	0	64	0	4913
14347	0	0	0	0	0	0	10436
14348	0	0	0	0	14	0	4224
14349	0	0	0	0	84	0	4221
14350	0	0	0	0	0	0	75220
14351	0	0	0	0	0	0	4864
14352	750	3100	0	0	810	0	5464
14353	0	0	0	0	22	0	4608
14354	0	0	0	0	0	0	42645
14355	0	0	0	0	0	0	3066
14356	0	0	0	0	0	0	16816
14357	0	0	0	0	0	0	8353
14358	0	0	0	0	0	0	3167
14359	0	0	0	0	96	0	3939
14360	0	0	0	0	0	0	1988
14361	0	0	0	0	692	0	14292
14362	0	0	0	0	0	0	1173
14363	295	695	0	0	389	0	570
14364	0	0	0	0	0	0	41791
14365	0	0	0	0	0	0	5854
14366	1340	9160	0	0	333	0	13123
14367	0	0	0	0	0	0	36720
14368	0	0	0	0	0	0	15736
14369	0	0	0	0	0	0	12022
14370	0	0	0	0	0	0	8630
14371	0	5000	0	0	5512	0	4083
14372	0	0	0	0	7	0	1757

```
14373    490    8120     0     0   674      0   8151
14374      0       0     0     0     0      0   2413
14375      0       0     0     0     0      0   1378

[14376 rows x 141 columns]
```

In [48]:
```
df.head()
```

Out[48]:
```
   STATE       IDCENSUS                            NAME  CONUM  CSA   CBSA  \
0      1  1500100100000  AUTAUGA COUNTY SCHOOL DISTRICT   1001    N  33860
1      1  1500200100000  BALDWIN COUNTY SCHOOL DISTRICT   1003  380  19300
2      1  1500300100000  BARBOUR COUNTY SCHOOL DISTRICT   1005    N      N
3      1  1500300200000    EUFAULA CITY SCHOOL DISTRICT   1005    N      N
4      1  1500400100000     BIBB COUNTY SCHOOL DISTRICT   1007  142  13820

   SCHLEV   NCESID  YRDATA    V33  ...    V32    _19H   _21F   _31F    _41F  \
0       3  0100240      15   9664  ...      0   49431  16603   2992   63042
1       3  0100270      15  30596  ...      0  337160  99087  13027  423220
2       3  0100300      15    925  ...      0    8024      0    304    7720
3       3  0101410      15   2829  ...      0       0      0      0       0
4       3  0100360      15   3357  ...      0   22155      0   1190   20965

   _61V  _66V   W01    W31    W61
0     0     0  2094    372   8617
1     0     0  5784  50441  71370
2     0     0     0      0    646
3     0     0     0   2054   7478
4     0     0  1397    790   5400

[5 rows x 141 columns]
```

In [49]:
```
df.shape
```

Out[49]:
```
(14376, 141)
```

In [50]:
```
df.describe()
```

Out[50]:
```
              STATE      IDCENSUS         CONUM        SCHLEV   YRDATA  \
count  14376.000000  1.437600e+04  14376.000000  14376.000000  14376.0
mean      26.801336  2.728090e+13  29838.158598      2.883139     15.0
std       13.894331  1.389514e+13  14753.492121      1.271649      0.0
min        1.000000  1.500100e+12   1001.000000      1.000000     15.0
25%       15.000000  1.550327e+13  18063.000000      3.000000     15.0
50%       27.000000  2.750320e+13  30063.000000      3.000000     15.0
75%       38.000000  3.850053e+13  41009.000000      3.000000     15.0
max       51.000000  5.150230e+13  56045.000000      7.000000     15.0

                V33      TOTALREV       TFEDREV           C14  \
count  14376.000000  1.437600e+04  1.437600e+04  14376.000000
mean    3374.682526  4.546951e+04  3.708038e+03    915.914858
std    14419.737037  2.590623e+05  2.021328e+04   5972.748111
min        0.000000  0.000000e+00  0.000000e+00      0.000000
25%      305.000000  5.241500e+03  2.920000e+02     54.000000
50%      979.500000  1.402300e+04  8.470000e+02    180.000000
75%     2744.250000  3.791375e+04  2.433500e+03    545.250000
```

```
max    995192.000000  2.543738e+07  1.307783e+06  379531.000000

                 C15      ...            V32          _19H          _21F  \
count   14376.000000      ...   14376.000000  1.437600e+04  1.437600e+04
mean      752.781093      ...       7.318656  2.871850e+04  4.831726e+03
std      3619.184513      ...     122.649299  1.841763e+05  2.652979e+04
min         0.000000      ...       0.000000  0.000000e+00  0.000000e+00
25%         0.000000      ...       0.000000  8.400000e+01  0.000000e+00
50%       118.000000      ...       0.000000  4.034000e+03  0.000000e+00
75%       532.000000      ...       0.000000  1.866350e+04  3.030000e+02
max    248209.000000      ...    7753.000000  1.372802e+07  1.312286e+06

                _31F          _41F           _61V           _66V  \
count   14376.000000  1.437600e+04   14376.000000   14376.000000
mean     3936.853158  2.957606e+04     487.162354     553.761547
std     20071.591744  1.894224e+05    3555.988118    6968.740454
min         0.000000  0.000000e+00       0.000000       0.000000
25%         5.000000  7.150000e+01       0.000000       0.000000
50%       410.000000  4.143500e+03       0.000000       0.000000
75%      1845.000000  1.912600e+04       0.000000       0.000000
max    731854.000000  1.447108e+07  173300.000000  700000.000000

                 W01            W31           W61
count   14376.000000   14376.000000  1.437600e+04
mean     1335.636825    3710.651920  9.091542e+03
std     10651.097318   20183.852905  3.261580e+04
min         0.000000       0.000000  0.000000e+00
25%         0.000000       0.000000  8.350000e+02
50%         0.000000       0.000000  2.708500e+03
75%       491.000000     626.250000  7.567500e+03
max    869643.000000  885058.000000  2.355662e+06

[8 rows x 137 columns]
```

In [51]:
```
df.columns
```

Out[51]:
```
Index(['STATE', 'IDCENSUS', 'NAME', 'CONUM', 'CSA', 'CBSA', 'SCHLEV', 'NCESID',
       'YRDATA', 'V33',
       ...
       'V32', '_19H', '_21F', '_31F', '_41F', '_61V', '_66V', 'W01', 'W31',
       'W61'],
      dtype='object', length=141)
```

In [52]:
```
useful = df[['STATE', 'YRDATA', 'TOTALREV', 'TFEDREV', 'TSTREV', 'TLOCREV']]
```

In [53]:
```
spending = df[['TCURELSC', 'TCURINST', 'TCURSSVC', 'TCUROTH' ]]
```

In [54]:
```
clean_useful = useful.dropna()
```

In [55]:
```
clean_useful.head()
```

Out[55]:

	STATE	YRDATA	TOTALREV	TFEDREV	TSTREV	TLOCREV
0	1	15	79665	7574	53244	18847
1	1	15	330317	23602	143282	163433
2	1	15	10519	2518	5632	2369
3	1	15	26076	3374	16048	6654
4	1	15	31825	3586	21687	6552

```
In [56]: clean_useful.describe()
```

```
Out[56]:                STATE   YRDATA      TOTALREV       TFEDREV        TSTREV  \
         count  14376.000000  14376.0  1.437600e+04  1.437600e+04  1.437600e+04
         mean      26.801336     15.0  4.546951e+04  3.708038e+03  2.105122e+04
         std       13.894331      0.0  2.590623e+05  2.021328e+04  1.125639e+05
         min        1.000000     15.0  0.000000e+00  0.000000e+00  0.000000e+00
         25%       15.000000     15.0  5.241500e+03  2.920000e+02  2.132500e+03
         50%       27.000000     15.0  1.402300e+04  8.470000e+02  6.516000e+03
         75%       38.000000     15.0  3.791375e+04  2.433500e+03  1.667925e+04
         max       51.000000     15.0  2.543738e+07  1.307783e+06  9.837509e+06

                     TLOCREV
         count  1.437600e+04
         mean   2.071025e+04
         std    1.367609e+05
         min    0.000000e+00
         25%    1.821000e+03
         50%    5.360000e+03
         75%    1.631175e+04
         max    1.429209e+07
```

```
In [57]: clean_spending = spending.dropna()
```

```
In [58]: clean_spending.head()
```

```
Out[58]:    TCURELSC  TCURINST  TCURSSVC  TCUROTH
         0     72872     44085     23217     5570
         1    269928    155668     99682    14578
         2      9957      5249      3835      873
         3     24232     14887      7494     1851
         4     29133     16019     10822     2292
```

```
In [59]: clean_spending.describe()
```

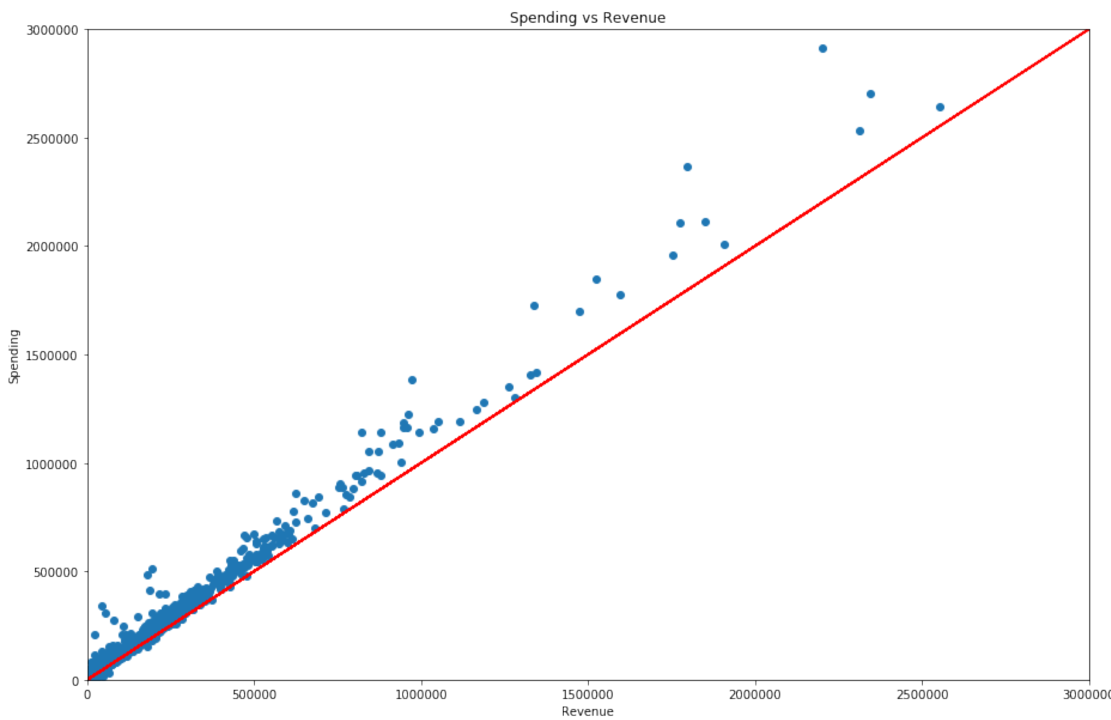
```
Out[59]:           TCURELSC      TCURINST      TCURSSVC        TCUROTH
         count  1.437600e+04  1.437600e+04  1.437600e+04   14376.000000
         mean   3.909932e+04  2.394922e+04  1.352417e+04    1625.927309
         std    2.401800e+05  1.777658e+05  5.894857e+04    7558.732209
         min    0.000000e+00  0.000000e+00  0.000000e+00       0.000000
         25%    4.437000e+03  2.508750e+03  1.577000e+03     170.000000
         50%    1.179900e+04  6.929000e+03  4.279500e+03     470.000000
         75%    3.228275e+04  1.913625e+04  1.153675e+04    1250.000000
         max    2.426924e+07  1.903582e+07  4.694906e+06  538505.000000
```

```
In [60]: spending = clean_useful['TOTALREV']
         revenue = clean_spending['TCURELSC']
```

```
In [61]: plt.figure(1)
         plt.figure(figsize=(15,10))
         plt.scatter(revenue, spending)
         myOLS_points = sm.OLS(revenue,revenue).fit()
         plt.plot(revenue, myOLS_points.predict(revenue), color = 'red')
         plt.title("Spending vs Revenue")
         plt.xlabel("Revenue")
         plt.xlim([0, 3*10e5])
```

```
plt.ylim([0, 3*10e5])
plt.ylabel("Spending")
plt.show()
plt.close()
```

<matplotlib.figure.Figure at 0x1160493c8>



In [62]:
```
clean_useful.head()
```

Out[62]:

	STATE	YRDATA	TOTALREV	TFEDREV	TSTREV	TLOCREV
0	1	15	79665	7574	53244	18847
1	1	15	330317	23602	143282	163433
2	1	15	10519	2518	5632	2369
3	1	15	26076	3374	16048	6654
4	1	15	31825	3586	21687	6552

In [63]:
```
cal = clean_useful.groupby('STATE')['TSTREV'].sum()[5]
cal
```

Out[63]: 42360470

In [64]:
```
mass= clean_useful.groupby('STATE')['TSTREV'].sum()[22]
mass
```

Out[64]: 6808436

In [65]:
```
new_mexico= clean_useful.groupby('STATE')['TSTREV'].sum()[32]
new_mexico
```

Out[65]: 2595682

In [66]:
```
ohio= clean_useful.groupby('STATE')['TSTREV'].sum()[36]
ohio
```

Out[66]: 10169760

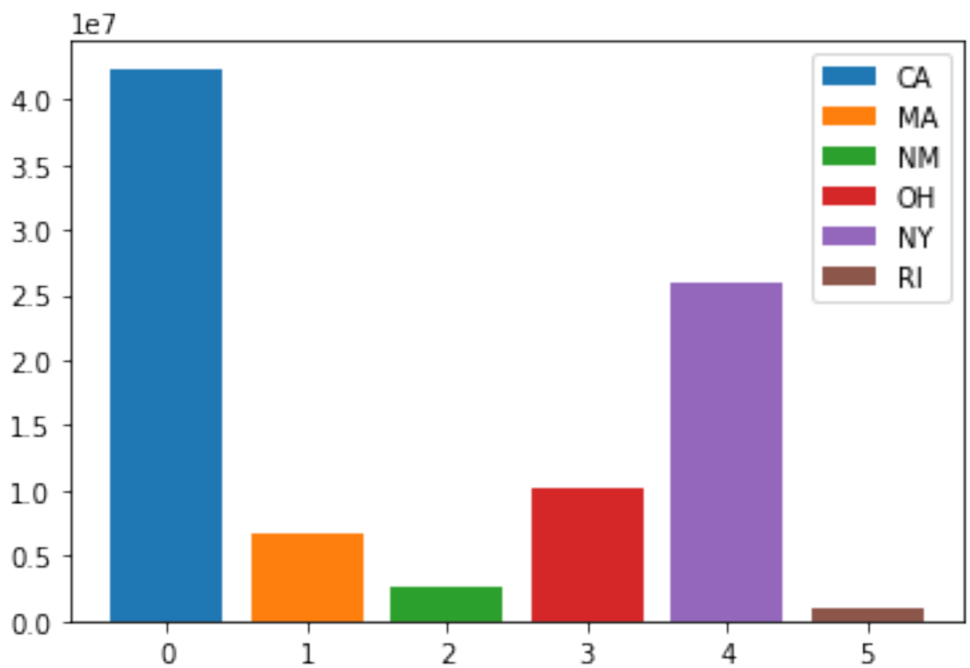
In [67]:
```
new_york= clean_useful.groupby('STATE')['TSTREV'].sum()[33]
new_york
```

Out[67]: 25900858

In [68]:
```
rhode_island= clean_useful.groupby('STATE')['TSTREV'].sum()[40]
rhode_island
```

Out[68]: 908963

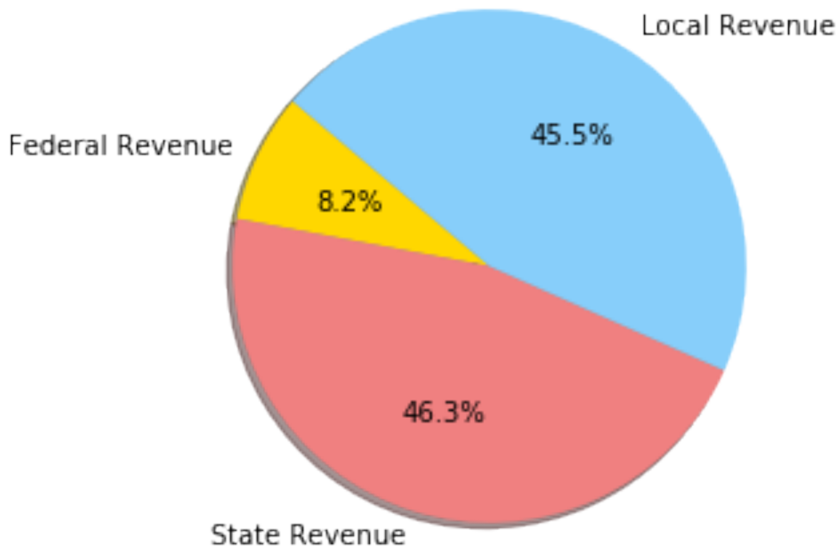
In [69]:
```
plt.bar(0, cal, label='CA')
plt.bar(1, mass, label='MA')
plt.bar(2, new_mexico, label='NM')
plt.bar(3, ohio, label='OH')
plt.bar(4, new_york, label='NY')
plt.bar(5, rhode_island, label='RI')
plt.legend()
plt.show()
```



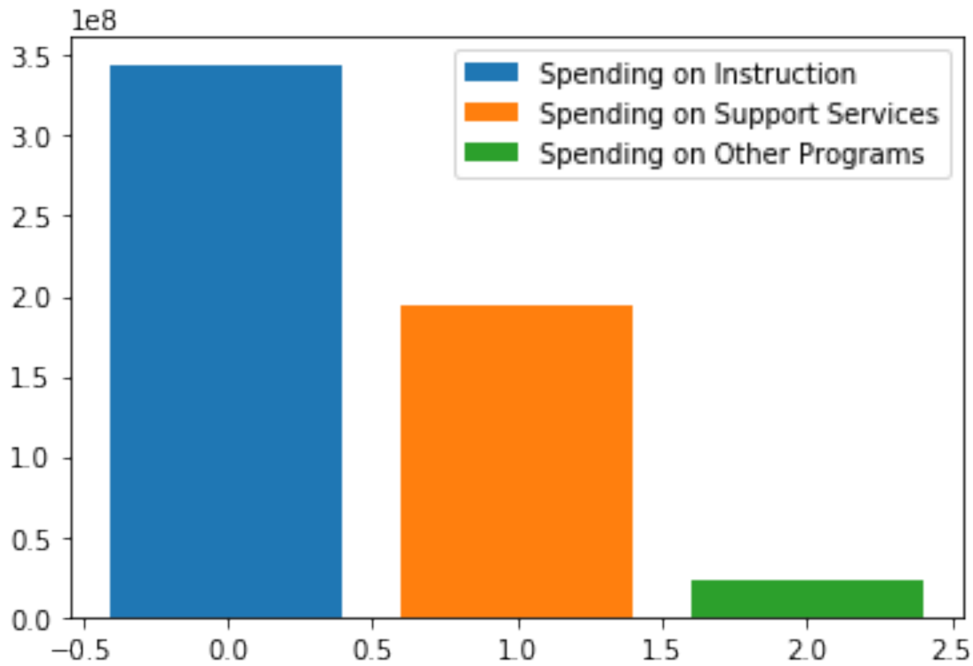
In [70]:
```
labels = 'Federal Revenue', 'State Revenue', 'Local Revenue'
sizes = [sum(clean_useful['TFEDREV']), sum(clean_useful['TSTREV']), sum(clean_useful['TLOCREV']
colors = ['gold','lightcoral', 'lightskyblue']
```

```
plt.pie(sizes, labels=labels, colors=colors,
        autopct='%1.1f%%', shadow=True, startangle=140)

plt.axis('equal')
plt.show()
```




```
In [76]: plt.bar(0, sum(clean_spending['TCURINST']), label='Spending on Instruction')
         plt.bar(1, sum(clean_spending['TCURSSVC']), label='Spending on Support Services')
         plt.bar(2, sum(clean_spending['TCUROTH']), label='Spending on Other Programs')
         plt.legend()
         plt.show()
```




```
In [ ]:
```

```
In [29]: !conda install seaborn --yes
```

```
Fetching package metadata ...
Solving package specifications: .

Package plan for installation in environment /Users/rishab/anaconda:

The following packages will be UPDATED:

    anaconda: 4.4.0-np112py36_0 --> custom-py36_0
    conda:    4.3.21-py36_0     --> 4.3.27-py36hb556a21_0
    seaborn:  0.7.1-py36_0      --> 0.8-py36_0

anaconda-custo 100% |################################| Time: 0:00:00   2.34 MB/s
conda-4.3.27-p 100% |################################| Time: 0:00:00   2.86 MB/s
seaborn-0.8-py 100% |################################| Time: 0:00:00   4.34 MB/s
```

```
In [30]: import seaborn as sns
```

```
In [ ]:
```

```
In [31]: cal = clean_useful.groupby('STATE')['TSTREV']
         rev = clean_useful['TOTALREV'][:1000].dropna()
         #print(rev)
         spend = clean_spending['TCURELSC'][:1000].dropna()
```

```
In [32]: rev = rev.reshape(1000, 1)
         rev = np.concatenate([rev, np.ones((rev.shape[0], 1))], axis=1)
```

```
/Users/rishab/anaconda/lib/python3.6/site-packages/ipykernel_launcher.py:1: FutureWarning: reshape is de
  """Entry point for launching an IPython kernel.
```

In [33]:
```
spend.size
```

Out[33]: 1000

In [34]:
```
from sklearn.linear_model import LogisticRegression
logistic = LogisticRegression()
logistic.fit(rev,spend)
```

Out[34]:
```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
          penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
          verbose=0, warm_start=False)
```

In [35]:
```
lr = LogisticRegression()
X = rev
y = spend
lr.fit(X, y)
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, penalty='l2', random_state=None, tol=0.0001)
prob = lr.predict_proba(X[0])
```

```
/Users/rishab/anaconda/lib/python3.6/site-packages/sklearn/utils/validation.py:395: DeprecationWarning:
  DeprecationWarning)
/Users/rishab/anaconda/lib/python3.6/site-packages/sklearn/linear_model/base.py:352: RuntimeWarning: ove
  np.exp(prob, prob)
```

In [36]:
```
prob[0][0]
```

Out[36]: 2.1723278860552369e-37

In [ ]: