Data Insights Take-Home: Clustering on Expenses and Revenues

Marianne C. Halloran October 14, 2017

The principal value of detailing the financial information obtained in Form 990 is to bring insight and arrive at data-backed conclusions about the NPOs, and their ability to garner financial support to continue operations. Here, The idea is that, by understanding how a NPO obtains revenue and spends its funds, we will be better poised to understand its efficacy. It also answers the questions of the financial strength of the NPO (its ability to attract resources, level of reserves, financial accountability, etc).

Data selection rationale and visualizations

First, I generate visualizations of the sources of income for 501(c)(3) NPOs, based on fundraising, campaign, membership, government grants, gifts and service revenues. This can provide insights into the income nature of the NPOs. Some NPOs can receive most of their funds from chargings fees, or through government grants. *To some individuals, this can often play an important factor in their donation decision*.

Similarly, I generate visualizations of the expenses, based on functional, service, management, and fundraising expenses. Individuals interested in NPOs can be interested in how the NPOs are spending most of its resources on program matters and not on management or fundraising, for example.

Net assets provide some indication of the level of resources the filer has to help support its activities in the future.

Moreover, compensation of its employees versus its income and expenditure can bring important information about the NPOs and their financial health and resource allocation.

Here, I perform a basic clustering for three features: ***total compensation***, **total income plus assets**, and ***total expenses***. The idea behind this selection would be to identify similar NPOs and the relationship between the three features: is there an inherent separation in the data?

Pre-processing and Clustering

Data was normalized using z-scores.

I chose ***z-score normalization*** although all data is given in dollars, these values necessarily comparable. Standardizing them using z-scores is a best-practice to give it equal weights by minimizing the error function using the Newton algorithm, i.e. a gradient-based optimization algorithm. Normalizing the data improves convergence of such algorithms.

Suggestions:

- Financial information is more meaningful if viewed over a period of several years, seeing how organizations can change over time. A single year's Form 990 provides only a snapshot in time.

In [15]:

```
#===============================================================
==#
# LIBRARIES
  #
#===============================================================
==#
from __future__ import print_function
import numpy as np
import pandas as pd
from scipy import stats
from sklearn import preprocessing
import matplotlib.pyplot as plt
import seaborn as sns; sns.set()
from sklearn.cluster import KMeans
from sklearn import metrics
from scipy.spatial.distance import cdist
from pylab import rcParams
from mpl_toolkits.mplot3d import Axes3D

%matplotlib inline
rcParams['figure.figsize'] = 8,8
```

In [16]:

```
#===============================================================
==#
# DATA IMPPORT
  #
#===============================================================
==#
meta = pd.read_csv('input/NPO_meta_38k.csv')
meta.columns = ['EIN','contract_term','tax_status','org_name','c
ity','state','tax_year',
                'activity','year_formed','volunteer_ct','employe
e_ct','rev_campaigns',
                'rev_membership', 'rev_fundraising','rev_govgran
ts','rev_other','rev_progserv',
                'rev_netfundraising','total_revenue','total_reve
nuePY','exp_grants','exp_progserv',
                'exp_management','exp_fundraising','total_expens
es','total_compensations',
                'comp_more100k', 'net_assets','pol_act','lob_ac
t','foreign_office',
                'foreign_fundraising','foreign_assist']
del meta['EIN'],  meta['contract_term']# meta['activity'],meta
```

```
['year_formed'],
print(u"\u0011",'Cleaned data, removed NaN')


# I'm removing any organization that is not a 501(c)(3) and any
 orgs with NaN in a row
meta = meta.dropna(axis=0,how='any')
meta_501c3 = meta.loc[meta['tax_status'] == 0]
del meta; meta = meta_501c3
meta
```

▶ Cleaned data, removed NaN

Out[16]:

	tax_status	org_name	city	state	tax_ye
1	0	KBL LLP	BROOKLYN	NY	2014
2	0	Davis & Deal CPAs	GLENDORA	CA	2014
3	0	CBIZ Tofias	NEWPORT	RI	2014
4	0	RAYMOND F BOOK & ASSOCIATES PA	DOVER	DE	2014
5	0	Larry D Sturgill CPA PC	WISE	VA	2014
6	0	MORGENSTERN WAXMAN ELLERSHAW	DETROIT	MI	2014
7	0	Douglass Mischley and Associates	ELK GROVE	CA	2014
8	0	Chek Tan and Company	SAN FRANCISCO	CA	2014
9	0	RUBINO AND COMPANY CHARTERED	ROCKVILLE	MD	2014
11	0	NEW HORIZON ACADEMY FOR EXCEPTIONAL STUDENTSINC	Ocala	FL	2013
12	0	ROBERTS ALEXONIS GROUP PLLC	Tucson	AZ	2014
13	0	MYTEAM TRIUMPH INC	ADA	MI	2014
14	0	Dittrich & Associates PLLC	Cincinnati	OH	2014
15	0	MITCHELL & CO PC	LEESBURG	VA	2014
16	0	ERICKSON DEMEL & CO PLLC	AUSTIN	TX	2014
17	0	MURPHY & MURPHY CPA LLC	WASHINGTON	DC	2014
18	0	SCHEULEN PATCHETT & EDWARDS PC	WARRENTON	VA	2014
19	0	Grace Tax Advisory Group LLC	North Fort Myers	FL	2014
20	0	BEREA ROTARY FOUNDATION INC	BEREA	OH	2014
21	0	Parmelee Poirier & Associates LLP	NEWPORT	RI	2014
22	0	ROBERT C ALARIO CPA PC	WORCESTER	MA	2014

	tax_status	org_name	city	state	tax_ye
23	0	HENDERSON HUTCHERSON & MCCULLOUGH PLLC	CHATTANOOGA	TN	2014
24	0	GARRIS AND COMPANY PC	CHARLOTTESVILLE	VA	2014
25	0	WILKE & ASSOCIATES LLP	WEXFORD	PA	2014
26	0	OTIS ATWELL	SOUTH BURLINGTON	VT	2014
28	0	Shafer & MacRae CPAs	TEMECULA	CA	2014
29	0	PSK LLP	IRVING	TX	2014
33	0	WEBSTER & KIRK PLLC	FRANKFORT	KY	2014
34	0	CORBETS & ASSOCIATES INC	CLEVELAND	OH	2014
35	0	Strand & Associates	Tacoma	WA	2014
...	...	...	...	...	...
38440	0	Dwight Nakata CPA CFPR	CERRITOS	CA	2014
38441	0	MARGARET MATTHEWS CPA PS	Seattle	WA	2014
38442	0	JM SOLUTIONS LLC	KLAMATH FALLS	OR	2014
38483	0	Abhishek R Agrawal	Fairfield	CA	2014
38484	0	OMEGA PSI PHI FRATERNITY NU OMICRON CHAPTER EC...	SOUTH OZONE PARK	NY	2013
38485	0	STEPHANIE ZILL	Los Angeles	CA	2014
38486	0	KARL HAISER CPA	FLINT	MI	2014
38487	0	EMILY A DEWALD EA	PORT TREVORTON	PA	2014
38488	0	RICHARD V RUDOLPH CPA	NEW YORK	NY	2014
38489	0	SECHLER CPA PC	SCOTTSDALE	AZ	2014
38490	0	WICKS BROWN WILLIAMS & CO	SEBRING	FL	2014
38491	0	HIRSCH OELBAUM BRAM HANOVER & LISKER CPA	BROOKLYN	NY	2014

	tax_status	org_name	city	state	tax_ye
38492	0	WESSEL & COMPANY CPAS	JOHNSTOWN	PA	2014
38493	0	LINDQUIST VON HUSEN & JOYCE LLP	FOSTER CITY	CA	2014
38494	0	PDM LLP	LONG BEACH	CA	2014
38495	0	JOHNSON LAMBERT LLP	RALEIGH	NC	2014
38496	0	ROSEN & FEDERICO	DENVER	CO	2014
38497	0	BOCK & ASSOCIATES LLP	EL PASO	TX	2014
38498	0	Chris Kitchens CPA	Marietta	GA	2014
38499	0	PALMETTO MOLLO MOLINARO & PASSARELLO LLP	Fort Lauderdale	FL	2014
38500	0	Robert J Iracane CPA	PARSIPPANY	NJ	2014
38501	0	BERRY DUNN MCNEIL & PARKER LLC	HANOVER	MA	2014
38502	0	SMITH DUKES & BUCKALEW LLP	MOBILE	AL	2014
38503	0	FUST CHARLES CHAMBERS LLP	NEW HARTFORD	NY	2014
38504	0	United Church Residences of Moundsville	Marion	OH	2014
38505	0	IRIZARRY RODRIGUEZ & CO CPA PSC	BAYAMON	PR	2014
38506	0	Dittrick & Associates Inc	Chagrin Falls	OH	2014
38507	0	MATTHEWS CARTER & BOYCE	WASHINGTON	DC	2014
38508	0	WARNER & WARNER CPA'S INC	CARROLLTON	OH	2014
38509	0	Brown and Company	Washington	DC	2014

25244 rows × 31 columns

In [17]:

```
#================================================================
==#
# DESCRIPTIVE STATISTICS
  #
#================================================================
==#
print(u"\u0011",'Descriptive statistics, summarizing central ten
```

```
dency, dispersion')
print('  and shape of dataset\'s distribution')
meta.describe()
```

▶ Descriptive statistics, summarizing central tendency, dispersi
on
  and shape of dataset's distribution

Out[17]:

	tax_status	tax_year	year_formed	volunteer_ct	employee
count	25244.0	25244.000000	25244.000000	2.524400e+04	25244.00
mean	0.0	2013.980986	1220.854183	2.920713e+02	51.58572
std	0.0	0.136578	967.141939	1.449340e+04	477.3816
min	0.0	2013.000000	0.000000	0.000000e+00	0.000000
25%	0.0	2014.000000	0.000000	0.000000e+00	0.000000
50%	0.0	2014.000000	1972.000000	0.000000e+00	0.000000
75%	0.0	2014.000000	1997.000000	2.100000e+01	8.000000
max	0.0	2014.000000	2015.000000	2.000000e+06	36394.00

8 rows × 27 columns

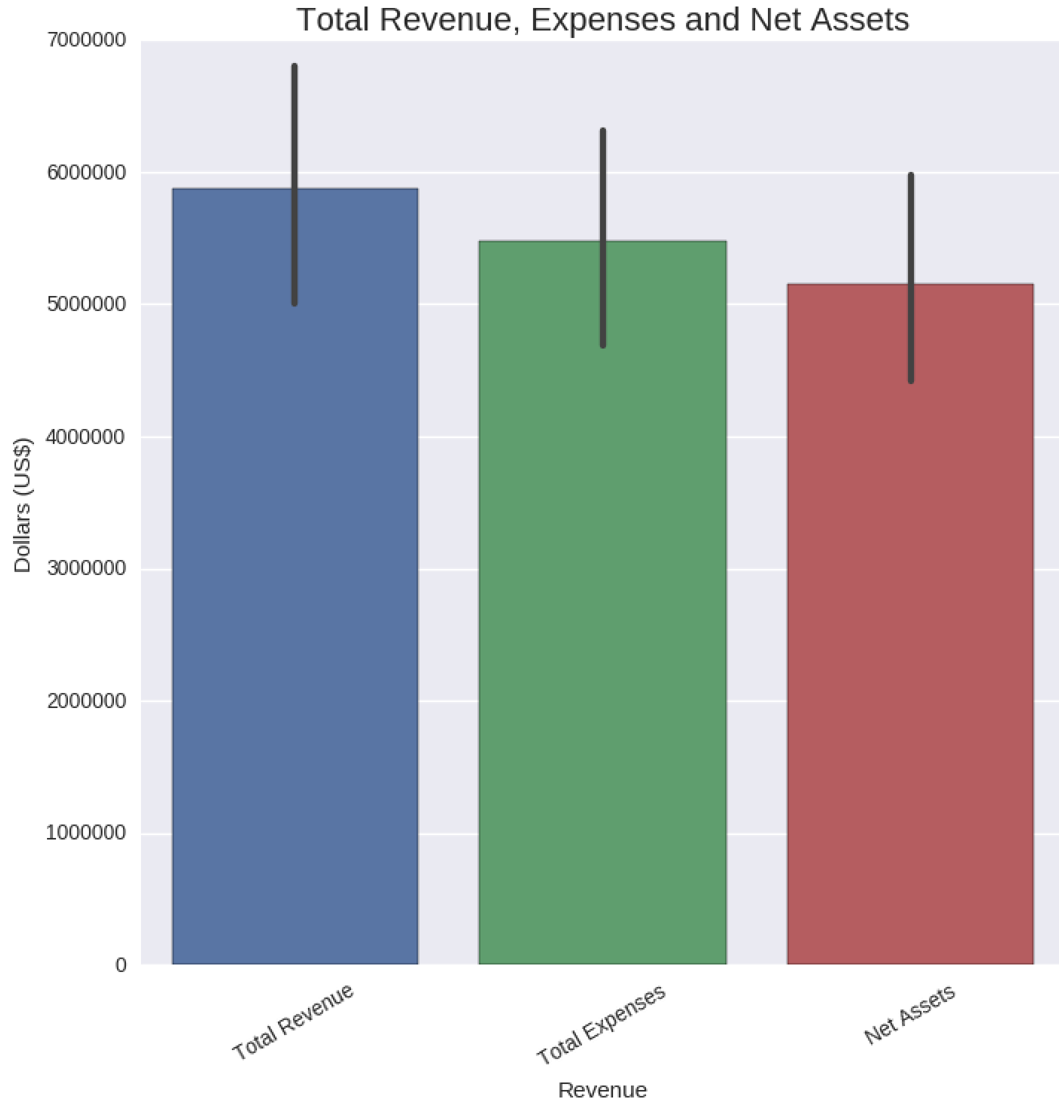
In [18]:

```
#=================================================================
==#
# PROCESS DATA: Categorical conversions, OHE, features
  #
#=================================================================
==#
# Cities and States will get categorical codes
meta['city'] = meta['city'].str.upper() # all upper case
cities = sorted(meta['city'].unique())  # sort by unique names
meta['city_int'] = meta['city'].map(lambda x: cities.index(x))
```

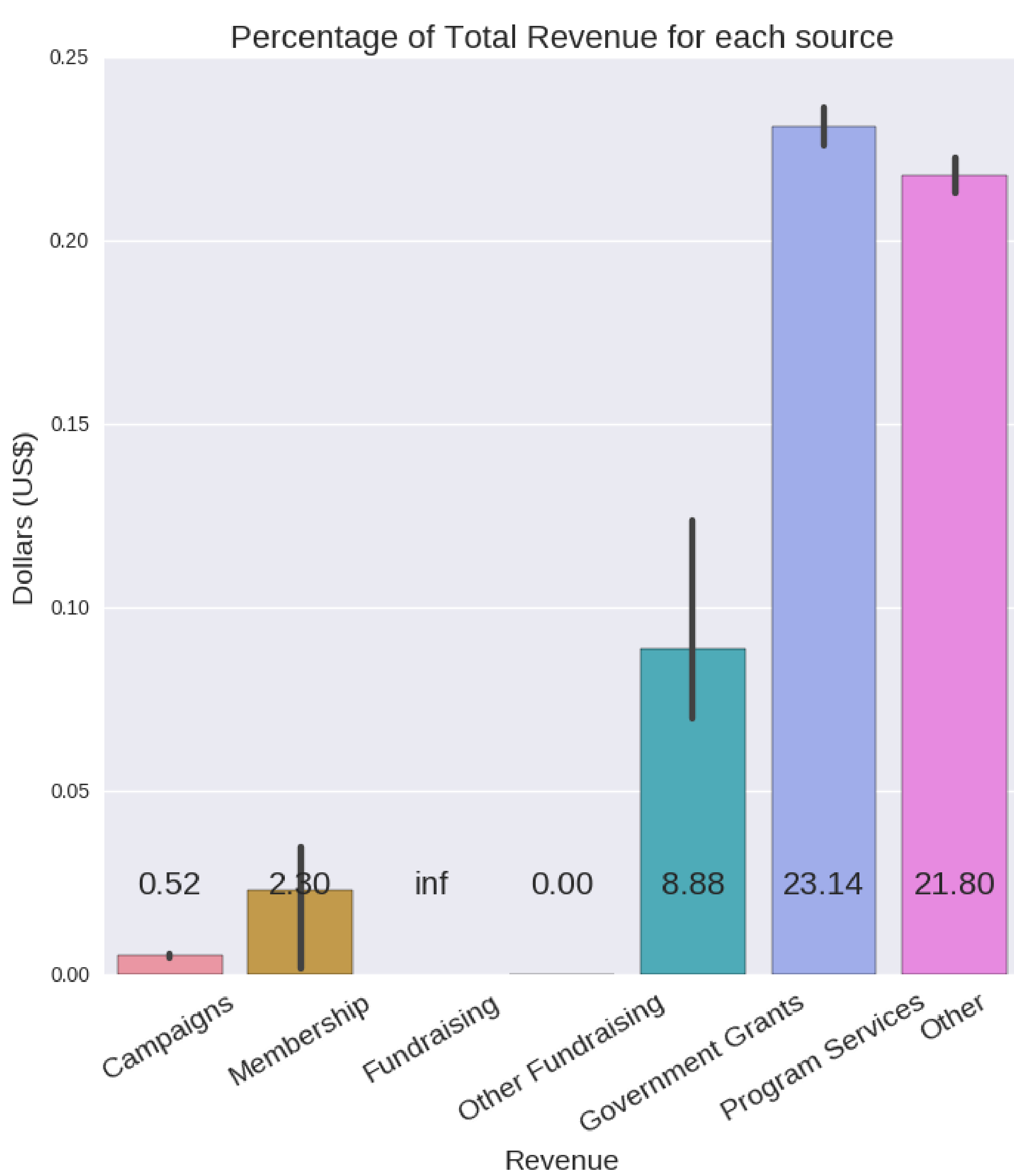
In [19]:

```
#=================================================================
==#
# VISUALIZATION
  #
#=================================================================
==#
# Visualizations of the sources of income for 501(c)(3) NPOs,
# based on fundraising, campaign, membership, government grants,

# gifts, assets and service revenues. This can provide insights
 into the
# income nature of the NPOs.
print(u"\u0011","It is interesting that in average, most NPOs ha
ve almost zero profitability (Income minus Expense)")
fig = plt.figure()
rev_df = meta[['total_revenue', 'total_expenses',
'net_assets']].copy()
ax = sns.barplot(data=rev_df)
ax.set(xlabel='Revenue', ylabel='Dollars (US$)')
ax.set_xticklabels(['Total Revenue','Total Expenses','Net Asset
s'], rotation=30)
ax.set_title('Total Revenue, Expenses and Net Assets',
fontsize=16)
plt.show()

del rev_df
# Revenue Plot, normalized by total revenue
print(u"\u0011","Note that most NPOs' income comes from Program
 Services (23%)",
      "followed closely by income from Other sources (Gifts, Don
ations,etc) at 21%. ",
      "Government Grants only account for 8.8% of the total reve
nue")
fig = plt.figure()
rev_df = meta[['rev_campaigns','rev_membership','rev_fundraisin
g','rev_netfundraising',

'rev_govgrants','rev_progserv','rev_other']].copy()

rev_df=(rev_df.div(meta['total_revenue'], axis=0)).fillna(0)
ax = sns.barplot(data=rev_df)
for p in ax.patches:
    ax.annotate("%.2f" % (p.get_height()*100),
                (p.get_x() + p.get_width() / 2., .02),
                fontsize=16,ha='center', va='bottom')
ax.set_xlabel('Revenue', fontsize=14)
ax.set_xticklabels(['Campaigns','Membership','Fundraising','Othe
r Fundraising',
                    'Government Grants', 'Program Services','Oth
er'], rotation=30, fontsize=14)
ax.set_title('Percentage of Total Revenue for each source', font
size=16)
ax.set_ylabel('Dollars (US$)',fontsize=14)
plt.show()
```

▶ It is interesting that in average, most NPOs have almost zero profitability (Income minus Expense)



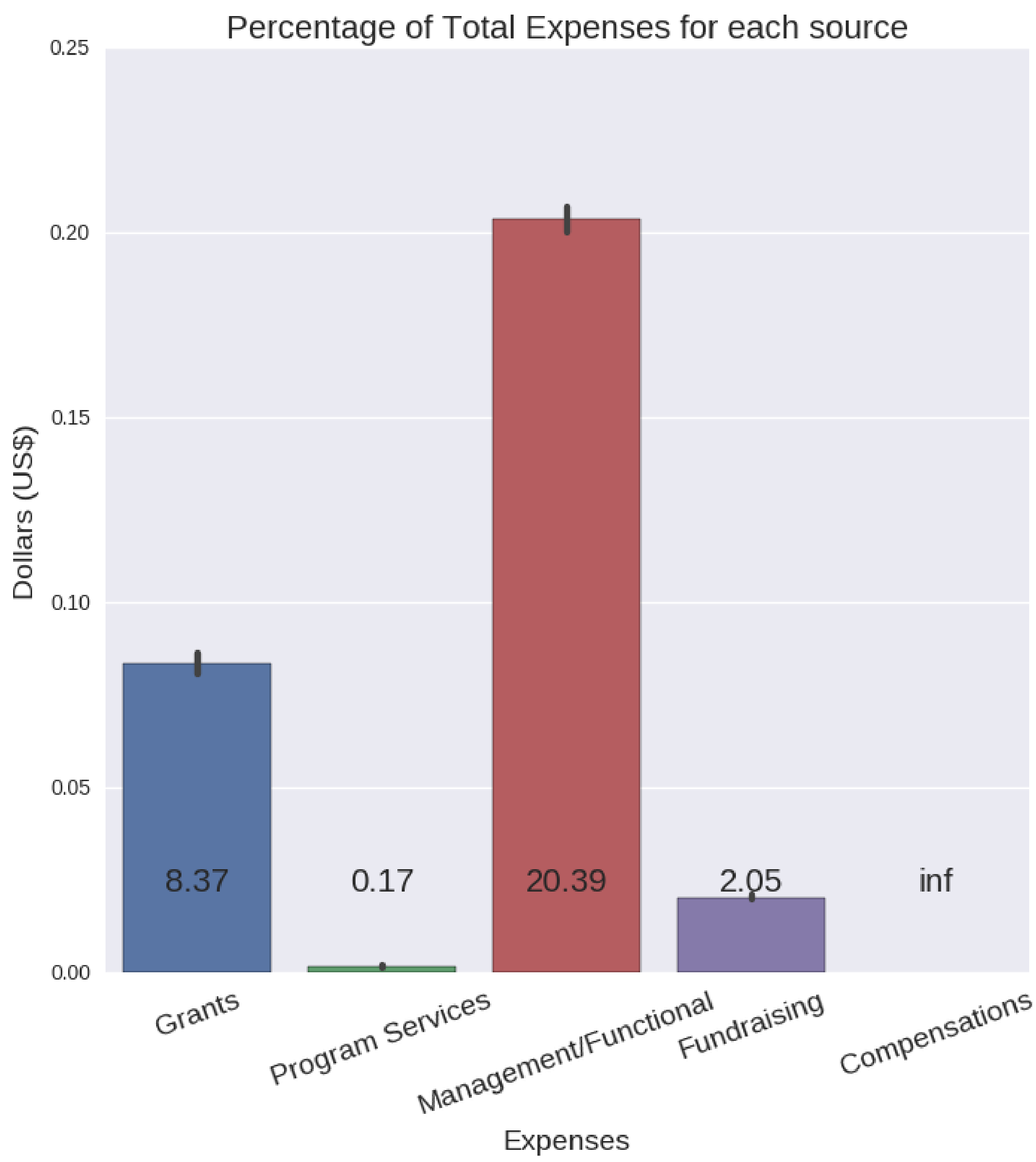
▶ Note that most NPOs' income comes from Program Services (23%) followed closely by income from Other sources (Gifts, Donations,etc) at 21%. Government Grants only account for 8.8% of the total revenue



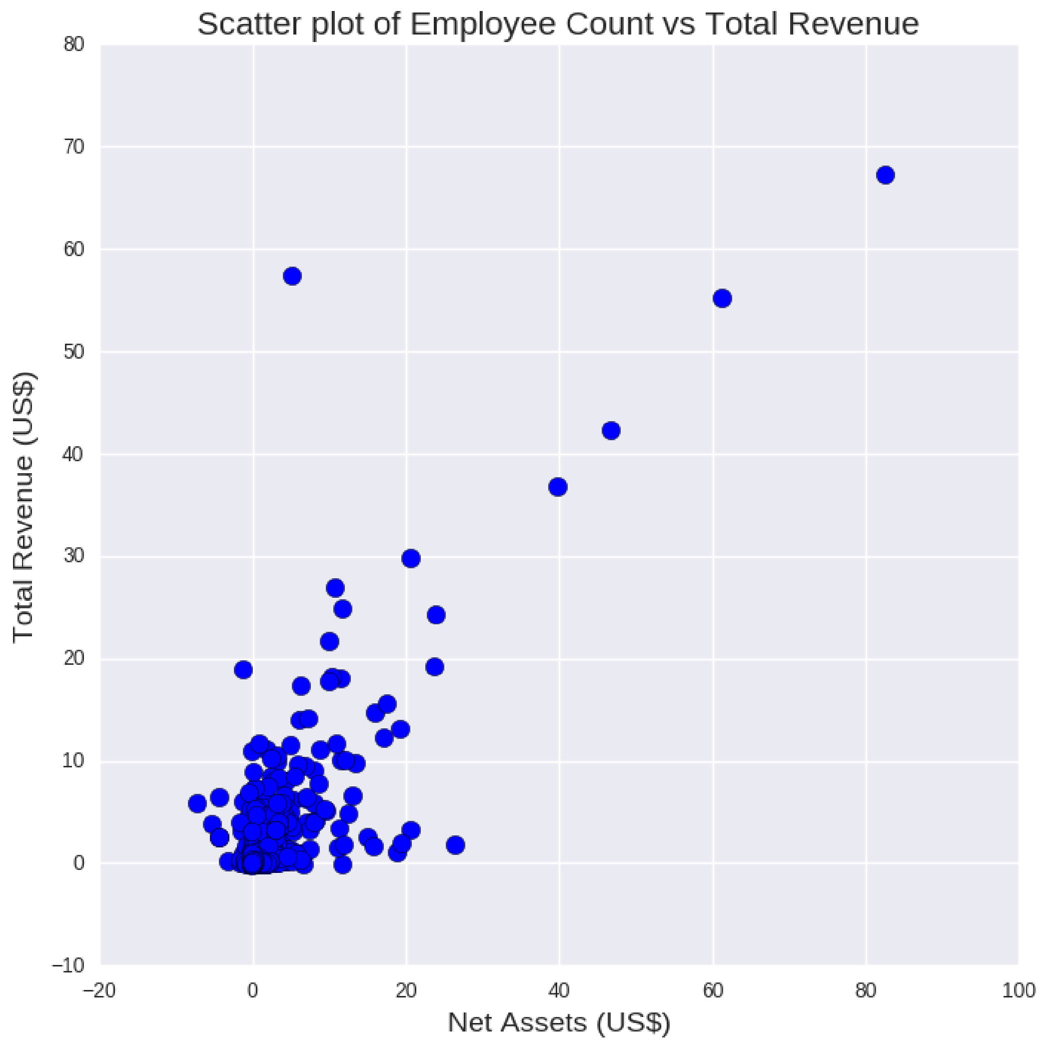
In [20]:
```
# Visualizations of the expenses, based on functional, service,
# management, and fundraising expenses.
# Normalized by total expenses
print(u"\u0011","Here, we see that, while Grants and Fundraising
 constitute only 8% of the expenses,",
      "Management/Functional and Compensations costs account, in
 average, for 22% of expenses.")
fig = plt.figure();
exp_df = meta[['exp_grants','exp_progserv','exp_management',
               'exp_fundraising', 'total_compensations']].copy()
exp_df=(exp_df.div(meta['total_expenses'], axis=0)).fillna(0)

ax = sns.barplot(data=exp_df)
```

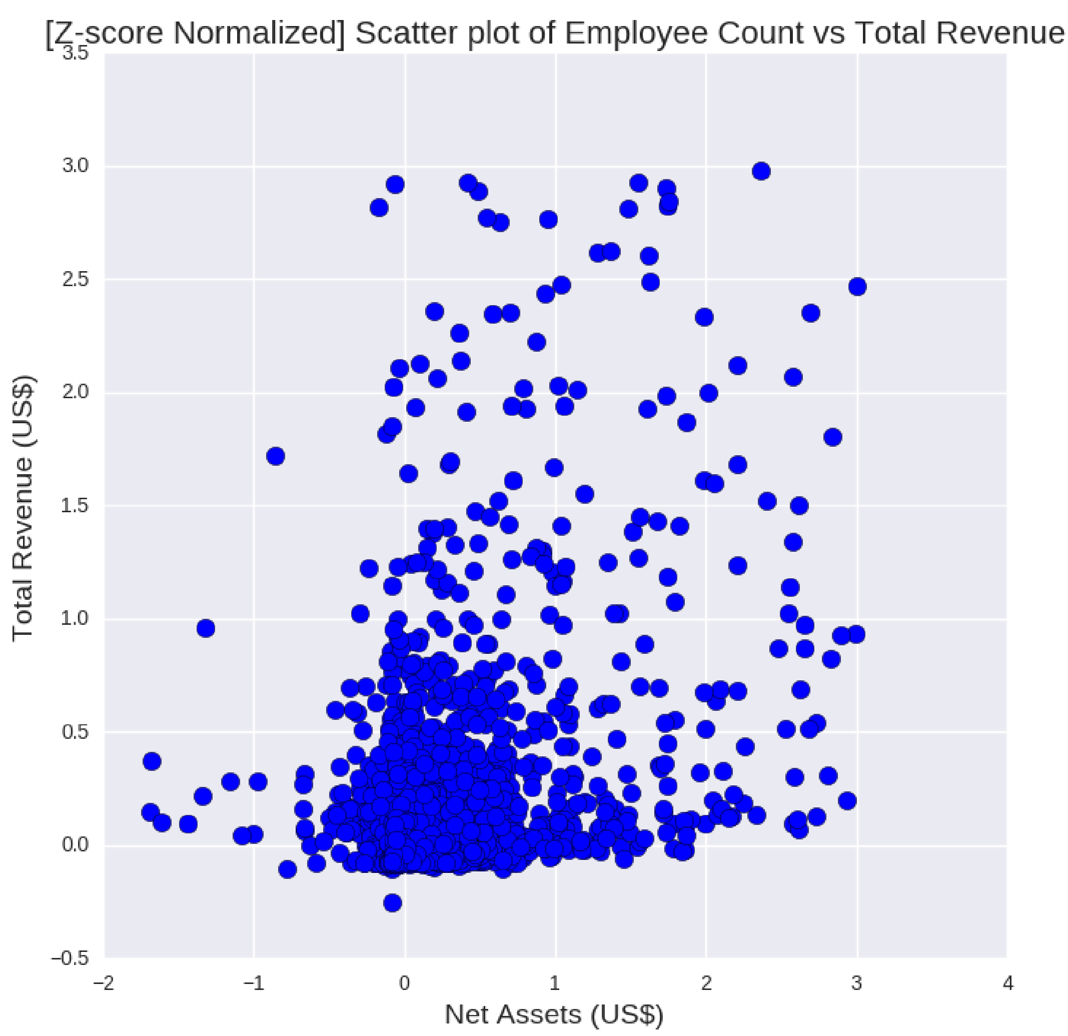
```
for p in ax.patches:
    ax.annotate("%.2f" % (p.get_height()*100),
                (p.get_x() + p.get_width() / 2., .02),
                fontsize=16,ha='center', va='bottom')
ax.set_xlabel('Expenses', fontsize=14)
ax.set_xticklabels(['Grants','Program Services','Management/Func
tional','Fundraising', 'Compensations'], rotation=20, fontsize=1
4)
ax.set_title('Percentage of Total Expenses for each source', fon
tsize=16)
ax.set_ylabel('Dollars (US$)',fontsize=14)
plt.show()
```

► Here, we see that, while Grants and Fundraising constitute only 8% of the expenses, Management/Functional and Compensations costs account, in average, for 22% of expenses.



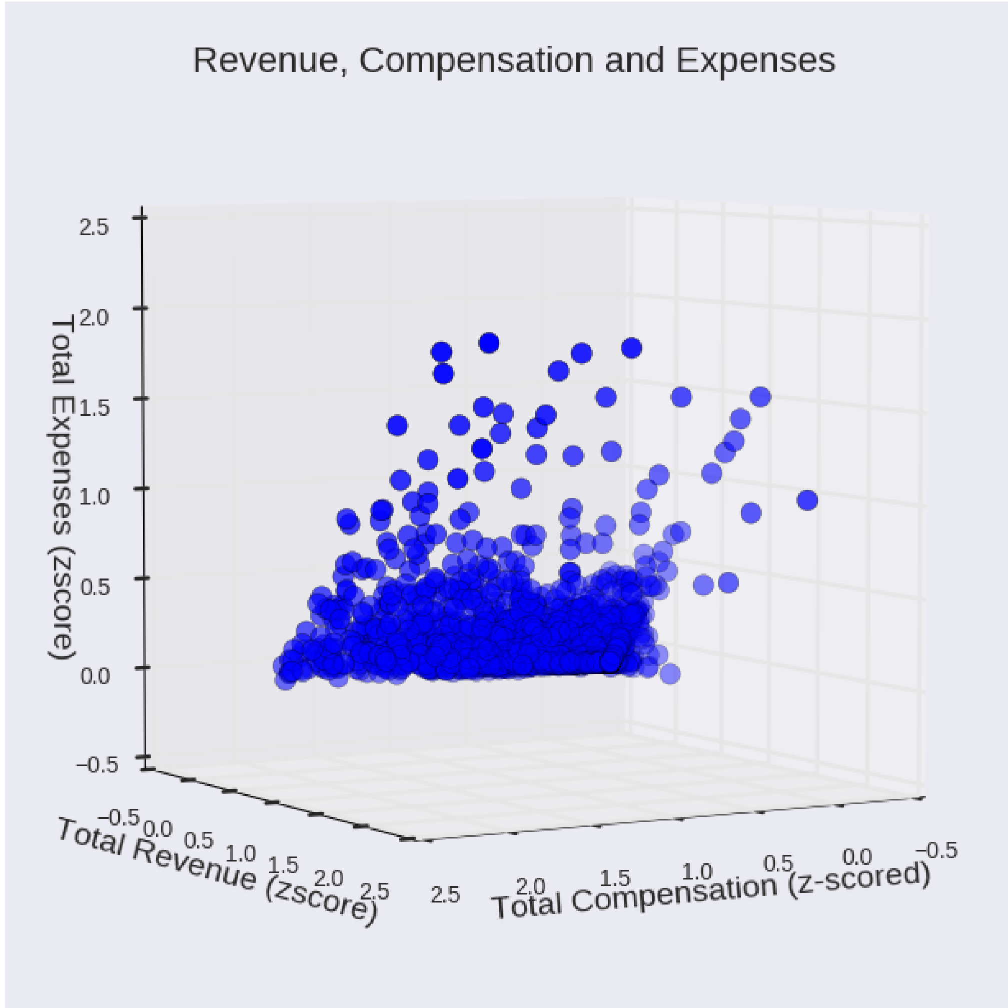
In [21]:
```
#=================================================================
==#
# VISUALIZATION
  #
#=================================================================
==#
## Create array for K-means
# Standarize (z-score) array (zi = xi-xmean/std)
meta_ = (meta[['net_assets', 'total_revenue']].copy()).apply(sta
ts.zscore)
meta_zscored = meta_[(np.abs(stats.zscore(meta_)) <
3).all(axis=1)]

## Visualizations
plt.scatter(meta_['net_assets'], meta_['total_revenue'], s=80);
plt.title('Scatter plot of Employee Count vs Total Revenue', fon
tsize=16)
plt.xlabel('Net Assets (US$)', fontsize=14); plt.ylabel('Total R
evenue (US$)', fontsize=14);
plt.show()

plt.scatter(meta_zscored['net_assets'], meta_zscored['total_reve
nue'], s=80);
plt.title('[Z-score Normalized] Scatter plot of Employee Count v
s Total Revenue', fontsize=16)
plt.xlabel('Net Assets (US$)', fontsize=14); plt.ylabel('Total R
evenue (US$)', fontsize=14);
plt.show()
```





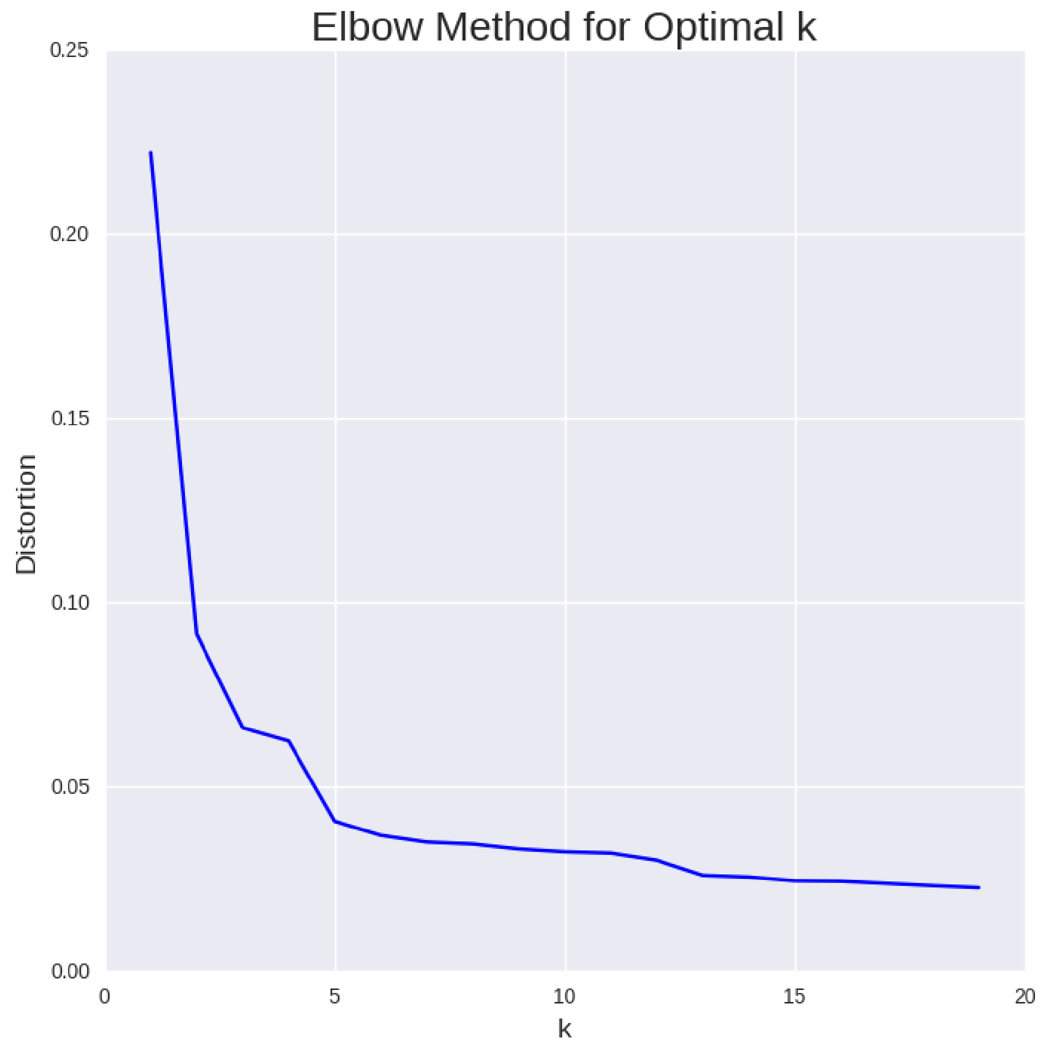
In [22]:

```
#================================================================
==#
# 3D VISUALIZATION
  #
#================================================================
==#
# Add columns City_int and State_int to processed data
meta_ = (meta[['total_compensations', 'total_revenue', 'total_ex
penses']].copy()).apply(stats.zscore)
meta_zscored = meta_[(np.abs(stats.zscore(meta_)) <
2).all(axis=1)]

# 3D Plot
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.scatter(meta_zscored['total_compensations'],meta_zscored['tot
al_revenue'],meta_zscored['total_expenses'], s=80)
ax.set_xlabel('Total Compensation (z-scored)', fontsize=14)
ax.set_ylabel('Total Revenue (zscore)', fontsize=14)
ax.set_zlabel('Total Expenses (zscore)', fontsize=14)
ax.set_title('Revenue, Compensation and Expenses', fontsize=16)
ax.view_init(elev=5., azim=60)
plt.show()
```



In [43]:

```
#==================================================================
==#
# K-Means Clustering
  #
#==================================================================
==#
# Elbow method to determine K
distortions = []
K = range(1,20)
for k in K:
    kmeanModel = KMeans(n_clusters=k).fit(meta_zscored)
    kmeanModel.fit(meta_zscored)
    distortions.append(sum(np.min(cdist(meta_zscored,
kmeanModel.cluster_centers_, 'euclidean'), axis=1)) / meta_zscor
ed.shape[0])
plt.plot(K, distortions, 'bx-')
plt.xlabel('k', fontsize=14)
plt.ylabel('Distortion', fontsize=14)
plt.title('Elbow Method for Optimal k', fontsize=20)
plt.show()

#==================================================================
==#
print(u"\u0011","Ideally, we would use a more stringent criterio
n determination method, such as",
      "the Akaike information criterion (AIC) or Bayesian inform
ation criterion (BIC)\n.")
# KMeans
kmeans = KMeans(n_clusters=4, random_state=0).fit(meta_zscored)
labels = kmeans.labels_
## Add labels to original data
meta_zscored = meta_zscored.assign(Clusters = labels)
columns = (meta_zscored.columns.get_values()).tolist()
print(u"\u0011","For k=4:",meta_zscored[columns].groupby(['Clust
ers']).mean())
```




```
▶ Ideally, we would use a more stringent criterion determination
method, such as the Akaike information criterion (AIC) or Bayesi
an information criterion (BIC)
.
▶ For k=4:            total_compensations  total_revenue  total_e
xpenses
Clusters
0                    -0.142374      -0.072098       -0.073396
1                     1.227261       0.164052        0.171487
2                     0.289131       0.032820        0.031737
3                     1.096792       1.097198        1.059560
```

In [44]:

```
#=================================================================
==#
# VISUALIZATION K-Means Clustering
  #
#=================================================================
==#
# Generate cluster groupings
cluster1=meta_zscored.loc[meta_zscored['Clusters'] == 0]
cluster2=meta_zscored.loc[meta_zscored['Clusters'] == 1]
cluster3=meta_zscored.loc[meta_zscored['Clusters'] == 2]
cluster4=meta_zscored.loc[meta_zscored['Clusters'] == 3]

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.scatter(cluster1['total_compensations'],cluster1['total_reven
ue'],cluster1['total_expenses'],
           c='blue', s=40, cmap="RdBu")
ax.scatter(cluster2['total_compensations'],cluster2['total_reven
ue'],cluster2['total_expenses'],
           c='red', s=40, cmap="RdBu")
ax.scatter(cluster3['total_compensations'],cluster3['total_reven
ue'],cluster3['total_expenses'],
           c='green', s=40, cmap="RdBu")
ax.scatter(cluster4['total_compensations'],cluster4['total_reven
ue'],cluster4['total_expenses'],
           c='yellow', s=40, cmap="RdBu")
ax.set_xlabel('Total Compensation (z-scored)', fontsize=14)
ax.set_ylabel('Total Revenue (zscore)', fontsize=14)
ax.set_zlabel('Total Expenses (zscore)', fontsize=14)
ax.set_title('KMeans Clustering for Revenue, Compensation and Ex
penses', fontsize=16)
ax.view_init(elev=5., azim=60)
plt.show()
```

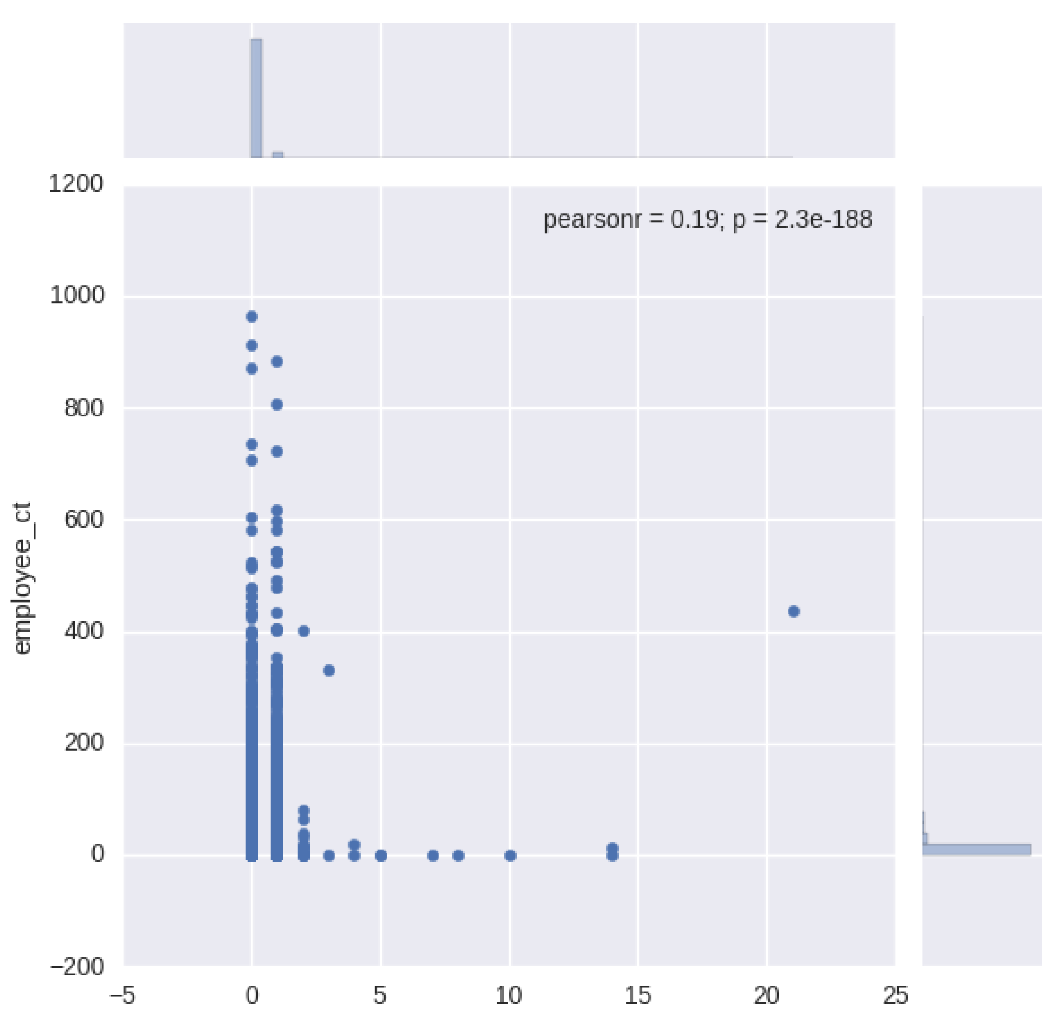
In [45]:

```
# Retrieve original data and clean NaNs due to zscore outlier re
moval
meta = meta.assign(Clusters = meta_zscored['Clusters'].loc[meta_
zscored.index.get_values()])
meta = meta.dropna(axis=0,how='any')
```

In [47]:

```
#================================================================
==#
# CLUSTER 1 ANALYSIS
  #
#================================================================
==#
print(u"\u0011","Cluster 1 contains ",len(cluster1),
"companies", "with an average of 9.84 Employees and Average Comp
ensation (Total) of US$19,137.95, with a mean of 0.049 employees
 receiving salaries above U$100k. ")
sns.jointplot(x="comp_more100k", y="employee_ct",
data=meta.loc[meta['Clusters'] == 0]); plt.show()
(meta.loc[meta['Clusters'] == 0]).describe()
```

```
► Cluster 1 contains  22960 companies with an average of 9.84 Em
ployees and Average Compensation (Total) of US$19,137.95, with a
mean of 0.049 employees receiving salaries above U$100k.
```



Out[47]:

	tax_status	tax_year	year_formed	volunteer_ct	emplo
count	22960.0	22960.000000	22960.000000	22960.000000	22960

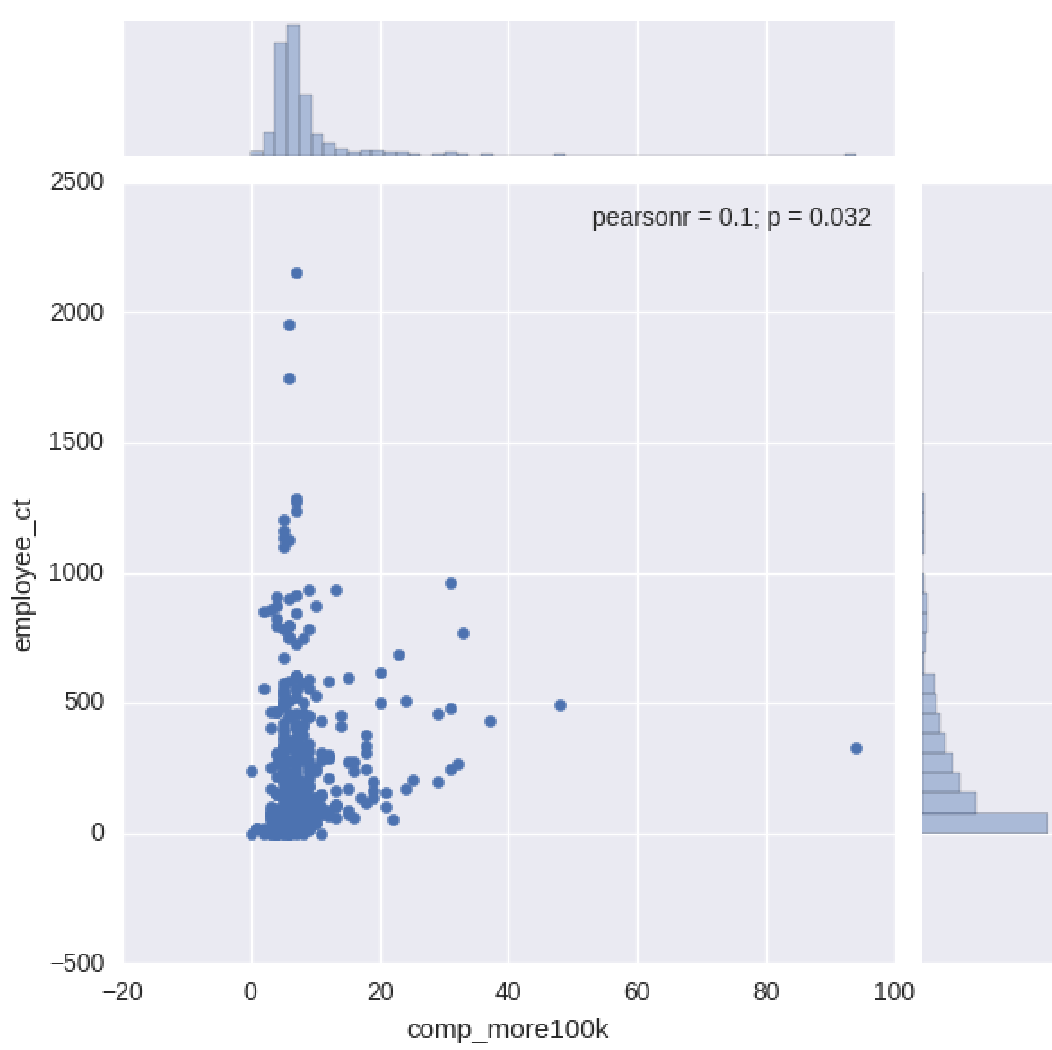
	tax_status	tax_year	year_formed	volunteer_ct	emplo
mean	0.0	2013.981533	1148.911237	83.683972	9.842(
std	0.0	0.134635	982.559518	1172.347520	38.282
min	0.0	2013.000000	0.000000	0.000000	0.000(
25%	0.0	2014.000000	0.000000	0.000000	0.000(
50%	0.0	2014.000000	1969.000000	0.000000	0.000(
75%	0.0	2014.000000	1998.000000	15.000000	4.000(
max	0.0	2014.000000	2015.000000	132183.000000	966.0(

8 rows × 29 columns

In [51]:

```
#==============================================================
==#
# CLUSTER 2 ANALYSIS
  #
#==============================================================
==#
print(u"\u0011","Cluster 2 contains",len(cluster2), "companies",
 "with an average of 246 Employees and Average Compensation (Tot
al) of US$1,023,683.00, with a mean of 7.96 employees receiving
 salaries above U$100k. ")
sns.jointplot(x="comp_more100k", y="employee_ct",
data=meta.loc[meta['Clusters'] == 1]); plt.show()
(meta.loc[meta['Clusters'] == 1]).describe()
```

```
► Cluster 2 contains 418 companies with an average of 246 Employ
ees and Average Compensation (Total) of US$1,023,683.00, with a
 mean of 7.96 employees receiving salaries above U$100k.
```



Out[51]:

	tax_status	tax_year	year_formed	volunteer_ct	employee
count	418.0	418.000000	418.000000	418.000000	418.00000
mean	0.0	2013.973684	1969.983254	718.490431	246.61244
std	0.0	0.160265	101.174812	5800.562456	295.75199
min	0.0	2013.000000	0.000000	0.000000	0.000000
25%	0.0	2014.000000	1964.000000	0.000000	47.000000
50%	0.0	2014.000000	1982.000000	17.000000	145.00000
75%	0.0	2014.000000	1996.000000	136.000000	337.50000
max	0.0	2014.000000	2012.000000	108383.000000	2152.0000

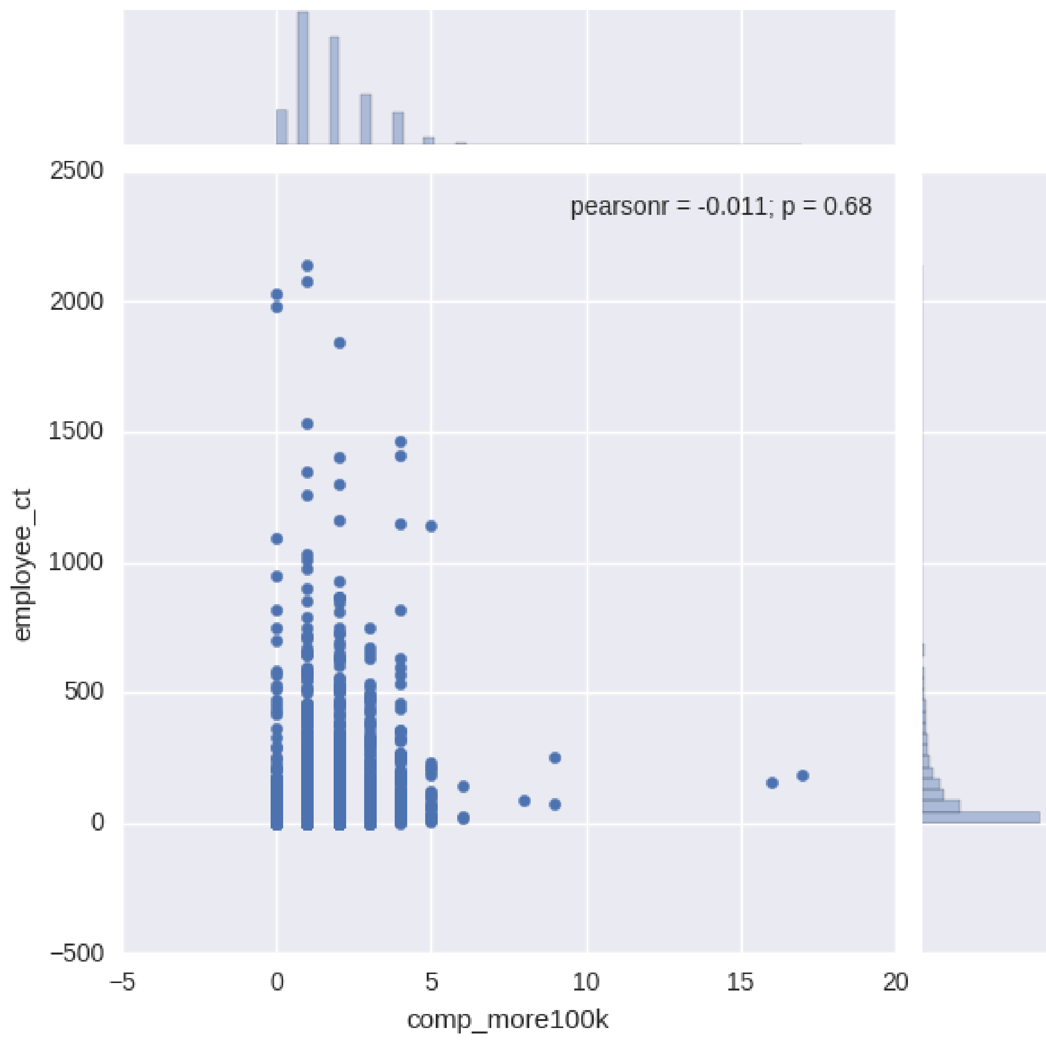
8 rows × 29 columns

In [52]:

```
#==============================================================
==#
# CLUSTER 3 ANALYSIS
  #
#==============================================================
==#
print(u"\u0011","Cluster 3 contains ",len(cluster3),
```

```
"companies", "with an average of 140 Employees and Average Compe
nsation (Total) of US$335,620.96, with a mean of 1.86 employees
 receiving salaries above U$100k. ")
sns.jointplot(x="comp_more100k", y="employee_ct",
data=meta.loc[meta['Clusters'] == 2]); plt.show()
(meta.loc[meta['Clusters'] == 2]).describe()
```

```
► Cluster 3 contains  1357 companies with an average of 140 Empl
oyees and Average Compensation (Total) of US$335,620.96, with a
 mean of 1.86 employees receiving salaries above U$100k.
```



Out[52]:

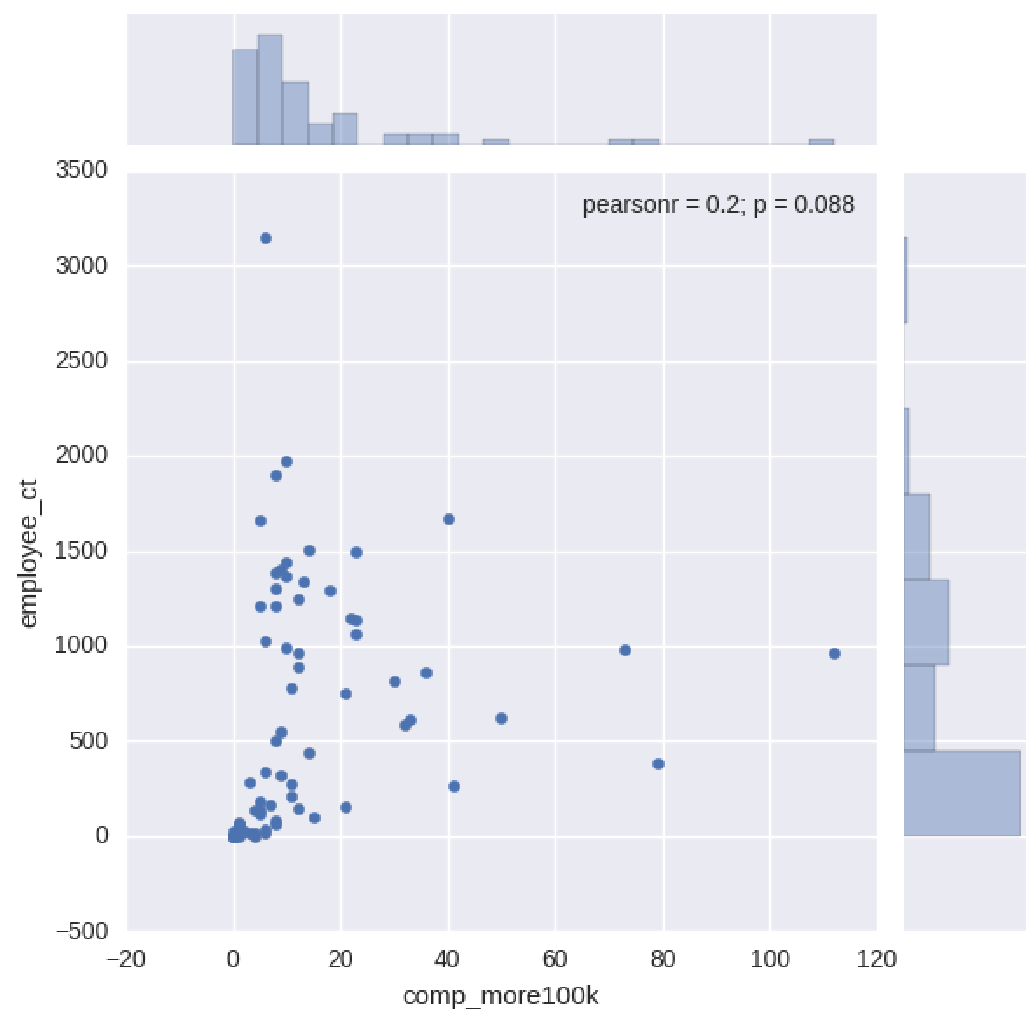
	tax_status	tax_year	year_formed	volunteer_ct	employee_
count	1357.0	1357.000000	1357.000000	1357.000000	1357.00000
mean	0.0	2013.977155	1937.671334	622.579956	140.24171
std	0.0	0.149463	277.662639	3540.320084	232.07243
min	0.0	2013.000000	0.000000	0.000000	0.00000
25%	0.0	2014.000000	1965.000000	0.000000	15.00000
50%	0.0	2014.000000	1983.000000	25.000000	52.00000
75%	0.0	2014.000000	1997.000000	200.000000	163.00000
max	0.0	2014.000000	2014.000000	85000.000000	2141.00000

8 rows × 29 columns

In [54]:

```
#================================================================
==#
# CLUSTER 4 ANALYSIS
  #
#================================================================
==#
print(u"\u0011","Similar to Cluster 3, Cluster 4 contains
",len(cluster4), "companies", "with an average of 647 Employees
 and Average Compensation (Total) of US$927,991.6e+05, with a me
an of 14.4 employees receiving salaries above U$100k. ")
sns.jointplot(x="comp_more100k", y="employee_ct",
data=meta.loc[meta['Clusters'] == 3]); plt.show()
(meta.loc[meta['Clusters'] == 3]).describe()
```

```
► Similar to Cluster 3, Cluster 4 contains  71 companies with an
average of 647 Employees and Average Compensation (Total) of US
$927,991.6e+05, with a mean of 14.4 employees receiving salaries
above U$100k.
```



Out[54]:

	tax_status	tax_year	year_formed	volunteer_ct	employee_
count	71.0	71.000000	71.000000	71.000000	71.000000
mean	0.0	2013.971831	1964.281690	1814.338028	647.323944
std	0.0	0.166633	37.291586	5377.264302	653.493115
min	0.0	2013.000000	1828.000000	0.000000	0.000000
25%	0.0	2014.000000	1943.500000	27.000000	67.500000
50%	0.0	2014.000000	1972.000000	94.000000	436.000000
75%	0.0	2014.000000	1992.500000	207.000000	1145.50000
max	0.0	2014.000000	2014.000000	28000.000000	3148.00000

8 rows × 29 columns

Results from Clustering

My analysis identified 4 clusters of companies in database. As shown above, Clusters 1 and 3 contain companies that are relatively small (9.84 and 140 employees in average, respectively), but counted with only, in average 0.049 and 1.86 of its employees receiving salaries over U$100,000, respectively.

On the other hand, Clusters 2 and 4 show NPOs with a comparatively larger number of employees (246 and 647 in average, respectively), however, its average number of employees receiving remuneration of U$100,000 and higher exceeds Clusters 1 and 3 117-fold (0.095 versus 11.18 average for Clusters 1,3 and Clusters 2,4, respectively). The relationship between the Clusters and their Compensations and Revenues can be seen in Result Figures 1 and 2, below.

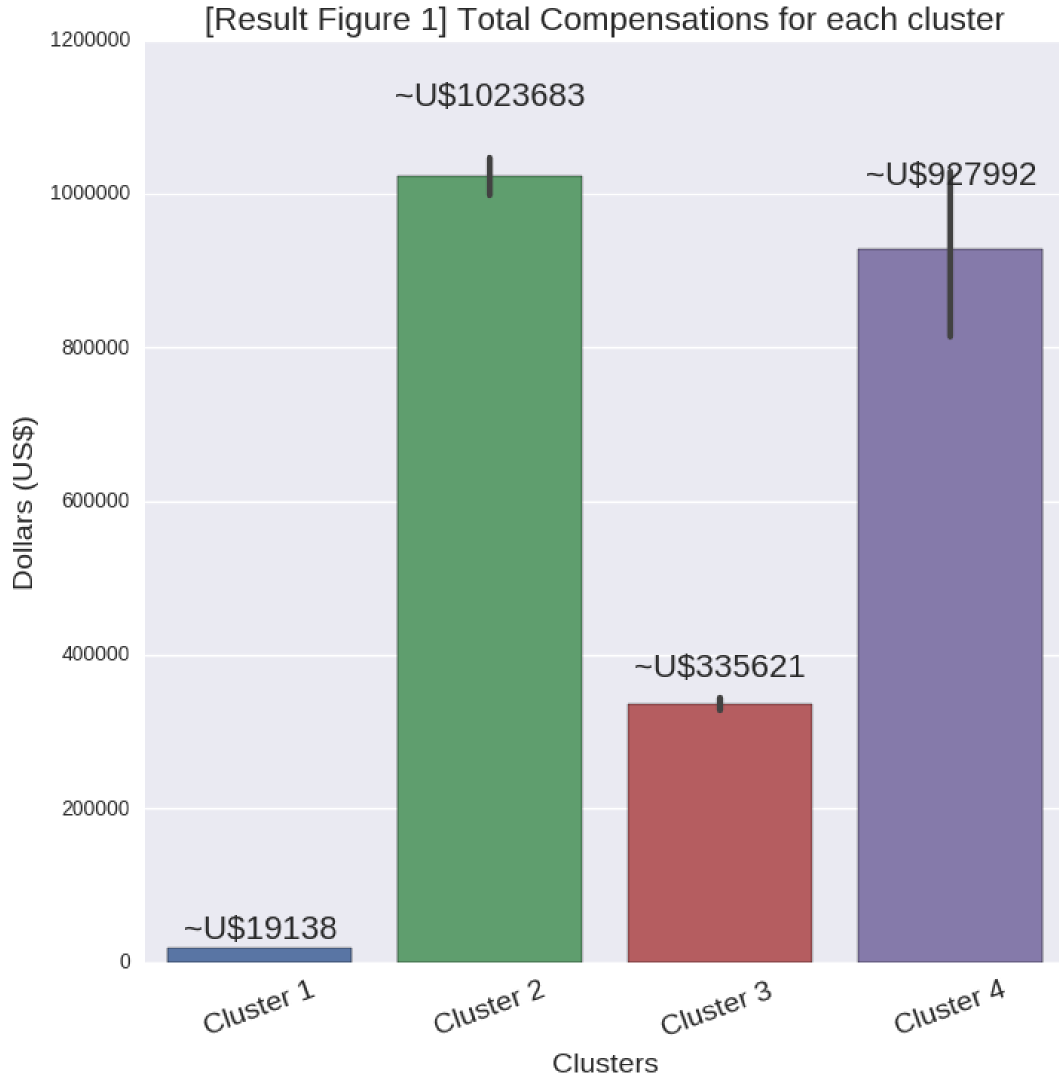
In Result Figure 2 and 3, I note that although NPOs in Cluster 4 are able to attract a larger amount of income, its number of volunteers varies greatly within the dataset, and is not much different from Clusters 2 and 3.

In addition, the three graphs below, show that revenues and expenses are also tied together for these clusters, and that the NPO clusters with highers revenues/expenses also count with the largest number of volunteers. Taken together, this data can help individuals and organizations best analyze the financial resources and its uses by the NPOs analyzed.

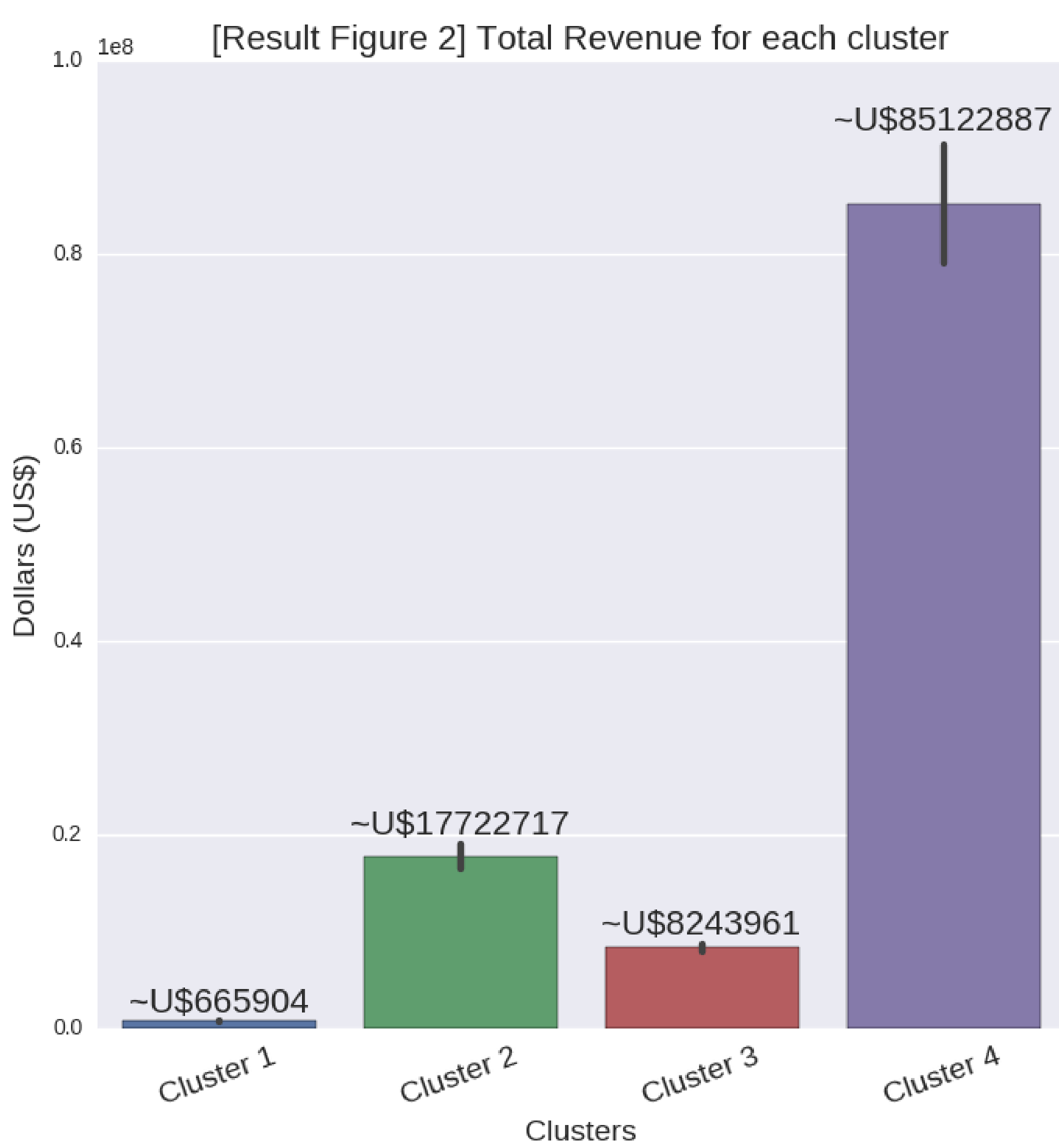
In [55]:

```
# Visualizations of the expenses, based on functional, service,
# management, and fundraising expenses.
# Normalized by total expenses
fig = plt.figure();
exp_df = meta[['total_compensations','Clusters']].copy()
exp_DF = exp_df.groupby(['Clusters'])
ax = sns.barplot(data=exp_df, x="Clusters", y="total_compensatio
ns", order=[0,1,2,3])
for p in ax.patches:
    ax.annotate("~U$%.0f" % (p.get_height()),
                (p.get_x() + p.get_width() / 2.,
p.get_height()*1.08),
                fontsize=16,ha='center', va='bottom')
ax.set_xlabel('Clusters', fontsize=14)
ax.set_xticklabels(['Cluster 1','Cluster 2', 'Cluster 3 ','Clust
er 4'], rotation=20, fontsize=14)
ax.set_title('[Result Figure 1] Total Compensations for each clu
ster', fontsize=16)
```

```
ax.set_ylabel('Dollars (US$)',fontsize=14)
plt.show()
```

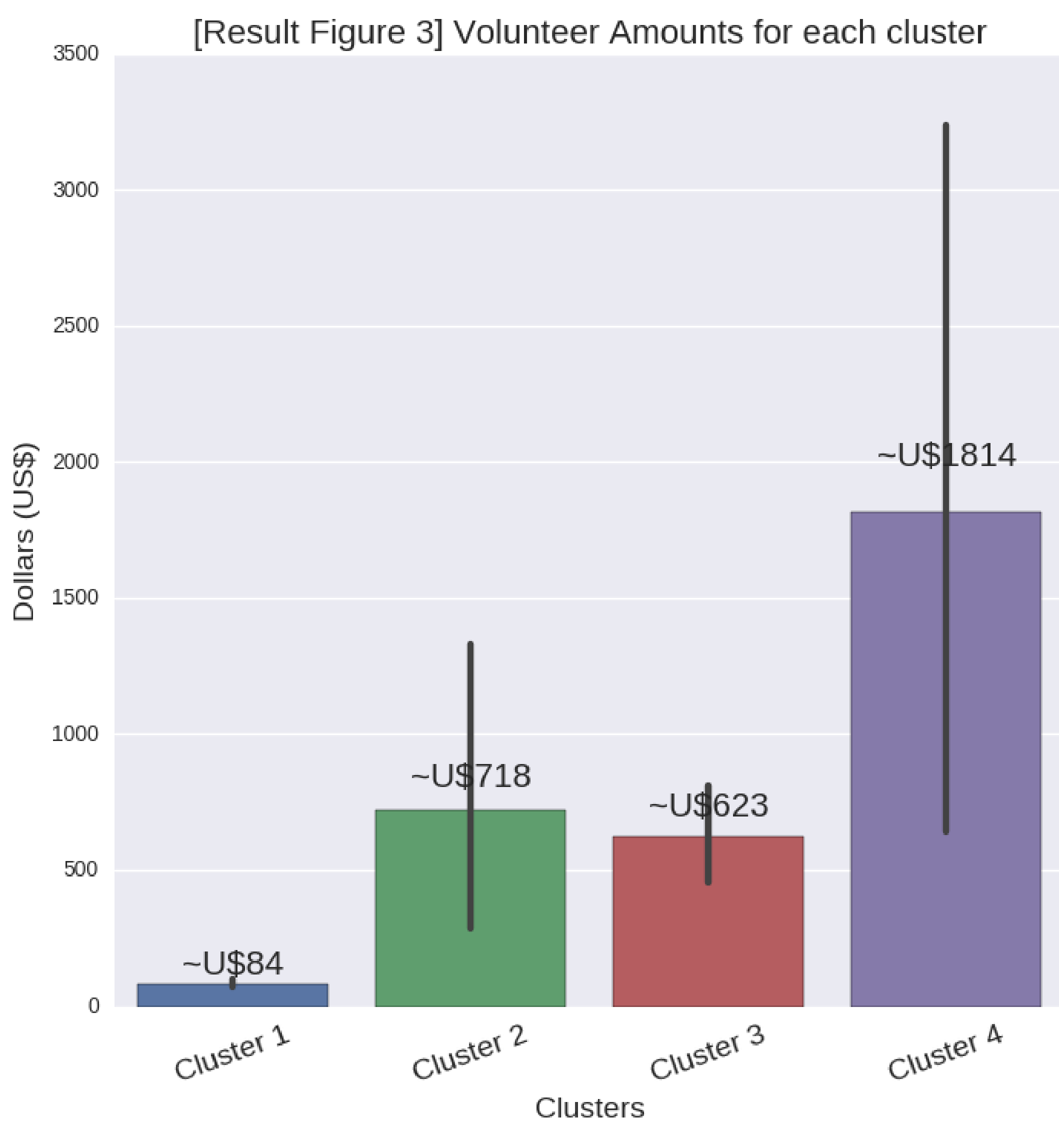


In [56]:
```
# Visualizations of the expenses, based on functional, service,
# management, and fundraising expenses.
# Normalized by total expenses
fig = plt.figure();
exp_df = meta[['total_revenue','Clusters']].copy()
exp_DF = exp_df.groupby(['Clusters'])
ax = sns.barplot(data=exp_df, x="Clusters", y="total_revenue", o
rder=[0,1,2,3])
for p in ax.patches:
    ax.annotate("~U$%.0f" % (p.get_height()),
                (p.get_x() + p.get_width() / 2.,
p.get_height()*1.08),
                fontsize=16,ha='center', va='bottom')
ax.set_xlabel('Clusters', fontsize=14)
ax.set_xticklabels(['Cluster 1', 'Cluster 2', 'Cluster 3','Clust
er 4'], rotation=20, fontsize=14)
ax.set_title('[Result Figure 2] Total Revenue for each cluster',
 fontsize=16)
ax.set_ylabel('Dollars (US$)',fontsize=14)
plt.show()
```



In [57]:

```
# Visualizations of the expenses, based on functional, service,
# management, and fundraising expenses.
# Normalized by total expenses
fig = plt.figure();
exp_df = meta[['volunteer_ct','Clusters']].copy()
exp_DF = exp_df.groupby(['Clusters'])
ax = sns.barplot(data=exp_df, x="Clusters", y="volunteer_ct", or
der=[0,1,2,3])
for p in ax.patches:
    ax.annotate("~U$%.0f" % (p.get_height()),
                (p.get_x() + p.get_width() / 2.,
p.get_height()*1.08),
                fontsize=16,ha='center', va='bottom')
ax.set_xlabel('Clusters', fontsize=14)
ax.set_xticklabels(['Cluster 1','Cluster 2', 'Cluster 3','Custe
r 4'], rotation=20, fontsize=14)
ax.set_title('[Result Figure 3] Volunteer Amounts for each clust
er', fontsize=16)
ax.set_ylabel('Dollars (US$)',fontsize=14)
plt.show()
```



In [ ]: