Section 3: Yelp Data Challenge - Clustering and PCA

Yiting Luo | Data Science Applied Research - 3

May 2018

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
% matplotlib inline
plt.style.use("ggplot")
```

In [2]:
```
df = pd.read_csv('dataset/last_2_years_restaurant_reviews.csv')
```

In [3]:
```
df.head()
```

Out[3]:

	business_id	name	categories	avg_stars	cool	date	funny	re
0	--9e1ONYQuAa-CB_Rrw7Tw	Delmonico Steakhouse	['Cajun/Creole', 'Steakhouses', 'Restaurants']	4.0	0	2016-03-31	0	6SgvNWJltnZhW7du
1	--9e1ONYQuAa-CB_Rrw7Tw	Delmonico Steakhouse	['Cajun/Creole', 'Steakhouses', 'Restaurants']	4.0	0	2015-06-29	0	iwx6s6yQxc7yjS7N
2	--9e1ONYQuAa-CB_Rrw7Tw	Delmonico Steakhouse	['Cajun/Creole', 'Steakhouses', 'Restaurants']	4.0	0	2015-03-16	0	UVUMu_bELdA56Ry
3	--9e1ONYQuAa-CB_Rrw7Tw	Delmonico Steakhouse	['Cajun/Creole', 'Steakhouses', 'Restaurants']	4.0	0	2016-02-10	0	UxFpgng8dPMWOj99
4	--9e1ONYQuAa-CB_Rrw7Tw	Delmonico Steakhouse	['Cajun/Creole', 'Steakhouses', 'Restaurants']	4.0	0	2017-02-14	0	Xp3ppynEvVu1KxDH

1. Cluster the review text data for all the restaurants

Define feature variables, here is the text of the review

```
In [4]: # Take the values of the column that contains review text data, save to a
        documents = df['text'].values
```

Define target variable (any categorical variable that may be meaningful)

For example, I am interested in perfect (5 stars) and imperfect (1-4 stars) rating

```
In [5]: # Make a column and take the values, save to a variable named "target"
        df['favorable'] = df['stars'] > 4
        target = df['favorable'].values
```

```
In [6]: target[:10]

Out[6]: array([ True, False,  True,  True,  True, False,  True,  True,  True,
               False])
```

look at the statistic of the target variable

```
In [7]: # To be implemented
        target.mean()

Out[7]: 0.46397299477268145
```

Create training dataset and test dataset

```
In [8]: from sklearn.cross_validation import train_test_split

        /Users/luoyiting/anaconda/lib/python3.5/site-packages/sklearn/cross_va
        lidation.py:44: DeprecationWarning: This module was deprecated in vers
        ion 0.18 in favor of the model_selection module into which all the ref
        actored classes and functions are moved. Also note that the interface
        of the new CV iterators are different from that of this module. This m
        odule will be removed in 0.20.
          "This module will be removed in 0.20.", DeprecationWarning)
```

In [9]:
```
# documents is X, target is y
# Now split the data to training set and test set
documents_train, documents_test, target_train, target_test = train_test_s
    documents, target, test_size = 0.95, random_state = 42
)
```

Get NLP representation of the documents

Fit TfidfVectorizer with training data only, then tranform all the data to tf-idf

In [10]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
```

In [11]:
```
# Create TfidfVectorizer, and name it vectorizer, choose a reasonable max
vectorizer = TfidfVectorizer(stop_words = 'english', max_features = 1000)
```

In [12]:
```
# Train the model with training data
vectors_train = vectorizer.fit_transform(documents_train).toarray() # toa
```

```
/Users/luoyiting/anaconda/lib/python3.5/site-packages/sklearn/feature_
extraction/text.py:1059: FutureWarning: Conversion of the second argum
ent of issubdtype from `float` to `np.floating` is deprecated. In futu
re, it will be treated as `np.float64 == np.dtype(float).type`.
  if hasattr(X, 'dtype') and np.issubdtype(X.dtype, np.float):
```

In [13]:
```
# Get the vocab of tfidf
words = vectorizer.get_feature_names()
```

In [14]:
```
# Use the trained model to transform all the reviews
vectors_documents = vectorizer.transform(documents).toarray()
```

```
/Users/luoyiting/anaconda/lib/python3.5/site-packages/sklearn/feature_
extraction/text.py:1059: FutureWarning: Conversion of the second argum
ent of issubdtype from `float` to `np.floating` is deprecated. In futu
re, it will be treated as `np.float64 == np.dtype(float).type`.
  if hasattr(X, 'dtype') and np.issubdtype(X.dtype, np.float):
```

Cluster reviews with KMeans

Fit k-means clustering with the training vectors and apply it on all the data

```
In [15]: from sklearn.cluster import KMeans
         kmeans = KMeans()
         kmeans.fit(vectors_train)

Out[15]: KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
             n_clusters=8, n_init=10, n_jobs=1, precompute_distances='auto',
             random_state=None, tol=0.0001, verbose=0)
```

Make predictions on all your data

```
In [16]: assigned_cluster = kmeans.predict(vectors_documents)
```

Inspect the centroids

To find out what "topics" Kmeans has discovered we must inspect the centroids. Print out the centroids of the Kmeans clustering.

These centroids are simply a bunch of vectors. To make any sense of them we need to map these vectors back into our 'word space'. Think of each feature/dimension of the centroid vector as representing the "average" review or the average occurances of words for that cluster.

```
In [17]: # cluster centers shape
         print ("cluster centers:")
         print (kmeans.cluster_centers_.shape)

         n_feat = 10
         top_centroids = kmeans.cluster_centers_.argsort()[:, -1:-n_feat:-1]

         cluster centers:
         (8, 1000)
```

```
In [18]: top_centroids.shape

Out[18]: (8, 9)
```

Find the top 10 features for each cluster.

For topics we are only really interested in the most present words, i.e. features/dimensions with the greatest representation in the centroid. Print out the top ten words for each centroid.

- Sort each centroid vector to find the top 10 features
- Go back to your vectorizer object to find out what words each of these features corresponds to.

In [19]:
```
print ("top features for each cluster:")
for num, centroid in enumerate(top_centroids):
    print ("%d: %s" % (num, ",".join(words[i] for i in centroid)))
```

```
top features for each cluster:
0: place,food,best,vegas,amazing,delicious,love,service,definitely
1: pizza,crust,good,place,great,slice,cheese,just,like
2: food,order,just,time,service,like,minutes,got,didn
3: chicken,fried,rice,good,food,ordered,place,sauce,like
4: great,food,service,place,amazing,awesome,friendly,staff,atmosphere
5: burger,fries,burgers,good,shake,cheese,place,great,food
6: good,food,really,place,service,nice,pretty,like,great
7: sushi,rolls,place,roll,ayce,good,great,service,eat
```

Try different k

If set k == to a different number, the top features will change?

In [20]:
```
# cluster to 6 clusters
kmeans = KMeans(n_clusters = 6)
kmeans.fit(vectors_train)
```

Out[20]:
```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
    n_clusters=6, n_init=10, n_jobs=1, precompute_distances='auto',
    random_state=None, tol=0.0001, verbose=0)
```

In [21]:
```
assigned_cluster = kmeans.predict(vectors_documents)

n_feat = 10
top_centroids = kmeans.cluster_centers_.argsort()[:, -1:-n_feat:-1]

print ("top features for each cluster:")
for num, centroid in enumerate(top_centroids):
    print ("%d: %s" % (num, ",".join(words[i] for i in centroid)))
```

```
top features for each cluster:
0: place,food,best,vegas,amazing,love,delicious,service,good
1: food,order,just,time,service,like,minutes,place,got
2: pizza,good,crust,place,great,slice,cheese,just,like
3: great,food,service,place,good,amazing,friendly,awesome,staff
4: good,chicken,food,really,ordered,like,place,rice,sauce
5: burger,fries,burgers,good,shake,cheese,place,great,food
```

Print out the rating and review of a random sample of the reviews assigned to each cluster to get a sense of the cluster.

In [22]:
```
# random select sample from each cluster, then display samples with ratin

for i in range(kmeans.n_clusters):
    cluster = np.arange(0, vectors_documents.shape[0])[assigned_cluster =
    sample_reviews = np.random.choice(cluster, 2, replace = False)
```

```python
    sample_reviews = np.random.choice(cluster, 2, replace = False)
    print ("=" * 10)
    print ("Cluster %d:" % i)
    for review_index in sample_reviews:
        print("  %s -  " % df.iloc[review_index]['stars']),
        print("%s" % df.iloc[review_index]['text'])
    print
```

```
==========
Cluster 0:
  5 -
We were planning on coming for dinner but the plan got changed and we
came here for lunch instead. I ordered the seafood pasta and it was so
yummy!!!!!!! The flavor is perfect with all the seafood I love! The pr
ice is not too night for such a nice restaurant! I will definitely com
e back for there steak next time in Vegas!
  5 -
Super fun. The food is amazing. Step outside your comfort zone and try
Blackout. You won't be disappointed.
==========
Cluster 1:
  4 -
Been at this place couple times and every time still like the first ti
me. Meat quality is perfect! even without marinated. You had my 5* but
your server messed up!!!!

Ordered a bottle of soju me both me and cousin, I.D. check! of course
no complain on that. I'm 22 and my cousin is 28. The server looked at
my I.D. and gave it back, then looked at my cousin's then surprisingly
asked him " You're older than him( who is me )?!?!? ". Deep down in my
heart, I died a little bit :( only a bottle of soju you dont have to m
ake me feel that way :(( .

Just a funny story I wanna share but love this place, will always come
back.
  3 -
A Chicago-based mob speakeasy, in Vegas? I was so there last Saturday
night for one of my very sporadic date nights with the husband. Locate
d in the old Hooters (yeah, I'm from Vegas, I know these things), I sa
w that they turned that titty-tastic 80's chicken wing place into a da
rk, somewhat intimate but very entertaining restaurant and lounge whic
h adorns pictures of mafia legends of long ago. Yup, this place defini
tely did not resemble the raucous orange splattered nipple nightmare o
f years before. I was impressed. We had 8:45 pm reservations and so up
on entering, you must ring the bell and low and behold, the host greet
s you from a small cut out in the wall above the payphone, to the righ
t of the door and doorbell. Woo, so speakeasy. And since we had those
reservations, dude had no problem letting us in. Don't know if a passw
ord is required without reservations, but one need only to check their
website for more info. But I think reservations would be most advisabl
e.

Anyway, it's a good thing we had a reservation because the place was p
acked. We had to wait a few minutes past our reservation time. The hos
```

```
t even admitted to us that he'd tell us to go to the bar, but consider
ing how packed it was that night, there was no way we were getting a s
eat at the bar. Once we were finally seated in a booth, not far from t
he entrance, we waited a little bit for a server. We weren't in a hurr
y, we didn't have any other plans or anywhere else to be that evening,
so to speak. But we were quite hungry and thirsty.

Having sorta kinda studied the menu before arriving, I chose the eggpl
ant parm and my husband tried the lasagna. We decided against any apps
that night, knowing that the entrees would be more than enough. The br
ead basket came and there were just 5 little slices of what looked lik
e baguette bread with cold individual foil wrapped pads of butter. I w
as not really impressed with that. If you're an Italian place, offer u
p warm breads and butter, not this airline bread and butter service! B
ut hey, it's always been said, best not to fill up on bread. So with t
hat, we waited for the cocktails. And because it was a speakeasy and I
knew any bar hopping that we may do that night would not offer good re
d wine, I opted for a dirty martini with blue cheese olives and it was
fabulous! The olives were perfect apps for me. The husband had his usu
al fancy outing drink: a Manhattan. And although dirty martinis go bet
ter with steak, I still enjoyed my eggplant parm with my drink. My hus
band raved about the lasagna, especially the sauce. We each took half
of our food home for the next day and it warmed up beautifully.

So I give it three stars only because of the pokey service and the bre
ad basket. I'd like to give this place another go, possibly take my mo
m here so that she can check it out. I might even entertain having a s
teak with my dirty martini next time I visit.
==========
Cluster 2:
  5 -
Stopped at the Cosmopolitan Hotel in Las Vegas to try the pizza at Sec
ret Pizza. It's not had to find, take the escalator to the 3rd floor f
ood area once on the 3rd floor you will see a small hallway with LP Al
bums on it follow it to some of the best pizza on the strip. The pizza
is sold by the slice($5.00) and. Cup of soda $4.00 (free refills while
eating there). The pizza tastes just like it should. It's not soggy an
d stands up to the last bite. I would recommend Secret Pizza to anyone
who wants to taste some of the best pizza around.
  3 -
Ehh more like 3.5 stars but closer to the three side than four side.

I come here when I'm looking for a quick and affordable meal in Vegas,
sober or drunk. It's open into the wee hours of the night which makes
it a great spot to come to after a fun night of Vegas shenanigans. Exp
ect a long wait if you're going to Vegas on a big holiday weekend or d
uring the summer though (I've had to wait close to 40 minutes in agoni
zing pain before...thanks, heels).

They offer a variety of pizzas, and I always go with some sort of meat
combo. It's served NY style and is a perfect thin crust. Slices come o
ut to be around $5 or more depending on the toppings your pizza has. A
ffordable for Vegas, but $5 for a slice of pizza is pretty ridiculous
```

```
elsewhere. Their slices are also pretty greasy which is a downside and
definitely doesn't sit too well especially when you're drunk off your
butt; delicious for the time being but regretful afterwards.

All in all, it's a good spot to come to when you're in Vegas and want
a decent slice of pizza. The fact that it's "secret" and is located th
rough a corridor also makes it a pretty cool spot.
==========
Cluster 3:
  4 -
Absolutely delicious! :) My friend and I got off of work and were look
ing for a new place to have a few drinks. I saw the Tacos and Beer sig
n and the appeal was instant.

The place itself has a great ambiance. Very welcoming and bartender wa
s friendly. The food was great. I had to order more.. I had the beef a
dobado and the green pork. I also had the beer batter fish. Yum!!!  Ma
rgarita was delicious! I love the salt foam with tajin. that adds a gr
eat kick to it. Nachos were not all that great though.

The only reason I did not give it 5 stars is due to the Happy Hour men
u was extremely limited. There is a great selection of beer and hoped
there would be more options on the HH menu. Just four taco options. I
had to order off the HH menu to get a variety of tacos. Also the barte
nders were friendly but the rest of the staff was not welcoming at all
. Sitting near the service bar, all I heard were the servers taking ba
d about their tables. I thought that was a bit unprofessional.

Regardless, I would absolutely recommend it to my friends!
  5 -
Great place for dinner. Lively atmosphere, but not too loud. Very clea
n. Pot roast was cooked perfectly and the mashed potatoes were great.
I wish more restaurants offered watermelon as a side because it was gr
eat...very sweet. The beer was perfect temp and carbonation. The waite
r was friendly and attentive. Will definitely come here more often.
==========
Cluster 4:
  5 -
My husband and I ordered the Gyro platter!! Absolutely delicious!! We
are from the east coast and wanted to try the restaurant based on the
reviews. The food was fresh and quickly served. The warm bread is the
best! I will return before we leave.

Day 2 - Just as good as Day 1. Great food!!
  4 -
My husband and I stopped in for a leisurely lunch after spending a rel
axing hour at the Float Centers of Nevada. If you're in the area, that
's a pretty good way to start your day!

We both love Thai food, and when it's our first time at a restaurant,
we order our favorite dishes to see how they stack up: shrimp pad thai
for me and beef pad see ew for him. Although my pad thai was too sweet
for my husband's tastes, that's the way I like it. The portion was hug
```

```
e, and the ingredients were in perfect balance. The beef pad see ew wa
s good, too; among the best my husband has had. They used Chinese broc
coli, which is always better than the more common florets, and the bee
f was tender, not chewy.

We also split an order of egg rolls, which were fine but unmemorable.
And I had a glass of the house Riesling, which was a little sweet as R
ieslings go but a good match for my spicy entree. Service was fine but
a bit slow, but since we were in no hurry to get anywhere, that was no
problem for us.

Next time we're up for a float, I think we will float on over to Archi
's for lunch. And by the looks of their weekday lunch specials (10 tas
ty selections, each with soup and egg roll), I think I'll try for a Fr
iday next time out!
==========
Cluster 5:
  4 -
Order the pork buns and the oxtail chili cheese fries because they're
amazing. The pork belly buns have thick cuts of pork and egg and the o
xtail chili cheese fries are piled high topped with a sweet and savory
oxtail chili cheese topping and jalapenos. The burgers are decent but
these two items on the menu always get me and I end up not ordering an
ything else. For hamburgers, I prefer Umami Burger, but their beef is
high quality from the burgers I've tried. It's just that no hamburger
particularly stands out enough for me to recommend.
  3 -
Went here twice cause there's really nothing better in the area. Chick
en pot pie was good, just a little dry. Coffee was bad (really weak).
Cappuccino was bad (too much foam, and also too watery). DeLux Burger
was average - did get medium rare right more or less. Good sauce for t
he fries. Everything in Vegas is overpriced so can't really complain a
bout the price.
```

2. Cluster all the reviews of the most reviewed restaurant

Find the most reviewed restaurant and analyze its reviews

In [23]:

```
# Find the business who got most reviews, get filtered df, name it df_top
most_reviewed_restaurant = df['business_id'].value_counts().index[0]

df_top_restaurant = df[df['business_id'] == most_reviewed_restaurant].cop

df_top_restaurant.head()
```

Out[23]:

	index	business_id	name	categories	avg_stars	cool	date	funny	
0	228947	RESDUcs7fliihp38-d6_6g	Bacchanal Buffet	['Sandwiches', 'Buffets', 'Restaurants', 'Brea...	4.0	0	2015-02-25	0	pWnR3FNc
1	228948	RESDUcs7fliihp38-d6_6g	Bacchanal Buffet	['Sandwiches', 'Buffets', 'Restaurants', 'Brea...	4.0	0	2017-07-02	0	Xci8QJaDC
2	228949	RESDUcs7fliihp38-d6_6g	Bacchanal Buffet	['Sandwiches', 'Buffets', 'Restaurants', 'Brea...	4.0	0	2015-12-29	0	IZO6o6I-
3	228950	RESDUcs7fliihp38-d6_6g	Bacchanal Buffet	['Sandwiches', 'Buffets', 'Restaurants', 'Brea...	4.0	0	2016-03-19	0	KNYBYYO
4	228951	RESDUcs7fliihp38-d6_6g	Bacchanal Buffet	['Sandwiches', 'Buffets', 'Restaurants', 'Brea...	4.0	0	2015-10-14	0	0E

We can also load restaurant profile information from the business dataset

In [24]:

```
# Load business dataset

import json
import pandas as pd

file_business = "dataset/business.json"
with open(file_business) as f:
    df_business = pd.DataFrame(json.loads(line) for line in f)
```

```
In [25]: # Take a look at the most reviewed restaurant's profile
         df_business[df_business['business_id'] == most_reviewed_restaurant]
```

Out[25]:

	address	attributes	business_id	categories	city	hours	is_oper
119907	3570 S Las Vegas Blvd	{'WheelchairAccessible': True, 'OutdoorSeating...	RESDUcs7fIiihp38-d6_6g	[Sandwiches, Buffets, Restaurants, Breakfast &...	Las Vegas	{'Monday': '7:30-22:00', 'Tuesday': '7:30-22:0...	1

```
In [26]: df_business[df_business['business_id'] == most_reviewed_restaurant]['cate

Out[26]: array([list(['Sandwiches', 'Buffets', 'Restaurants', 'Breakfast & Brun
         ch', 'Food'])],
               dtype=object)
```

```
In [27]: df_business[df_business['business_id'] == most_reviewed_restaurant]['attr

Out[27]: array([{'WheelchairAccessible': True, 'OutdoorSeating': False, 'Restau
         rantsTakeOut': False, 'BusinessAcceptsCreditCards': True, 'Restaurants
         GoodForGroups': True, 'Alcohol': 'full_bar', 'WiFi': 'no', 'Restaurant
         sDelivery': False, 'RestaurantsAttire': 'casual', 'HasTV': False, 'Noi
         seLevel': 'average', 'BusinessParking': {'valet': True, 'lot': False,
         'validated': False, 'garage': True, 'street': False}, 'GoodForMeal': {
         'dessert': False, 'lunch': True, 'breakfast': False, 'dinner': True, '
         brunch': True, 'latenight': False}, 'Caters': False, 'GoodForKids': Tr
         ue, 'RestaurantsReservations': False, 'RestaurantsPriceRange2': 3, 'Am
         bience': {'touristy': False, 'casual': True, 'trendy': False, 'romanti
         c': False, 'intimate': False, 'classy': False, 'divey': False, 'upscal
         e': False, 'hipster': False}, 'BikeParking': False, 'RestaurantsTableS
         ervice': True}],
               dtype=object)
```

Vectorize the text feature

```
In [28]: # Take the values of the column that contains review text data, save to a
         documents_top_restaurant = df_top_restaurant['text'].values
         documents_top_restaurant

Out[28]: array(["Date & Time: Tuesday, 2/24/15 @ 2pm\n\nThe buffet was for two
         people at the price of $83 and some change. There is such variety when
         first stepping into the buffet. There are sections into seafood, ameri
         can, mexican, italian, chinese/japanese, and dessert. They have charcu
         terie with a selection of cheeses as well as a salad section. They off
         er all you can drink beer and mimosas for $22 (each)\n\nThe buffet hig
         hlight for me was the fresh juices and dessert. The food was good, I r
         ather settle for a buffet a boulder station for cheaper, than to go he
         re again. The food here was a little bland and I didn't really enjoy m
         ost of it. They do have a tapas style going for some of the food selec
         tions so you can try most of the items they have to offer, but it was
         tions so you can try most of the items they have to offer, but it was
         missing something -flavor. I decided to try this place out due to the
         numerous reviews on how great this place is, I would recommend it for
         you to try it at least once (the seafood was awesome). This place has

         amazing food presentation, but other than that, the lack of flavor in
         most of its items makes me steer away from here again.",
                "I'm staying at Caesar's Palace for 4th of July weekend and tod
         ay was my very first time visiting the long anticipated Bacchanal Buff
         et. I have to admit, with all the hype and 5 star positive reviews I h
         ave heard of this buffet (from friends, peers and other reviews) there
         was a heck of a lot to live up to. The result? Two thumbs up of approv
         al. Absolutely AMAZING.\nUpon entrance, I was blown away by the decor.
         Stunning. The restaurant utilizes glassware for much of their decor, w
         hich made for contemporary, yet comfortable ambiance, garnished with a
         n array of the most appetizing dishes I have ever seen at a buffet - a
         nd I have been to many around the USA, especially buffets in Las Vegas
         .\nI sampled everything. Starting with the Chinese food to the Mexican
         Sope (stuffed with zucchini flower and cheese) these were incredible -
         I could have eaten THREE. Next was the Wagyu Beef, Brisket, and the Su
         ckling Pig - not fatty, full of meat, skin was cripsy - perfectly done
         and the chef on the meat station was a hoot! Ended the meal with an aw
         esome selection of desserts. The dessert station - my favorite - where
         do I even begin? (salivate salivate) Most of the buffets in Vegas I've
         eaten at are overly sweet and taste like they've come out of a box. Th
         e desserts here are spot on. Gelato and Ice Cream selection is excelle
         nt. Some of the best ice creams I have ever tasted - lemon basil sorbe
         t- heart heart. Rice pudding was refreshing and tart. The coconut macr
         oons, Good Lord. For the holiday weekend they even decorated some of t
         he desserts with little flags - super patriotic and thoughtful.\nOvera
         ll, I give this buffet 5 stars, A+++. Service was spot on as well, Sup
         er attentive. \nGreat attention to detail. \nIf you want a great meal,
         great dessert, and great service hit the Bacchanal. My only gripe was
         that the wait was long to get in, but 100% worth it.",
                'Great buffet and beautiful food presentation. I came for brunc
         h on Christmas Holiday and there was no line, so that was probably the
         highlight of the eatery. I wouldn\'t pay $70 for this buffet again bei
         ng that I had to wait in line for 15-20minutes for warm crab legs. The
         y need to find a better method for serving the most demanded item in t
         heir buffet. I say try it once but don\'t expect to get your food "rig
         ht away" at this buffet.',
                ...,
                'Arrive before 11 to avoid the crowd.\nWe walked right in when
         we got there around 10am on a Wednesday, but saw the huge line as we w
         ere leaving. Place was pretty good in terms of a buffet, wide selectio
         n, from salads, pizza, sushi, eggs. Other than my coffee mug being dir
         ty and having to ask for a different one, we enjoyed our breakfast.',
                "I have never been to another Vegas buffet so I have no real co
         mparison.  I did think the $40 something price tag for lunch was a lot
         at first, but once I saw everything included I didn't think it was bad
         .  I wish they had some alcohol options included though.  I way over i
         ndulged here, but everything was amazing.  Some of my favorites were t
         he Asian area and the carving station.  The carving station had everyt
         hing you could think of prime rib, ham, sausages, brisket, etc.  There
         was also the best sides near there great potatoes and vegetables.  I w
```

```
as a little disappointed in the seafood.  I hoped to see king or snow
crab legs, but they only had Jonah crab claws which I am not much a fa
n of and I prefer my crab warm.  The Asian section had dim sum, standa

rd Chinese dishes and sushi.  All of it great.  \n\nMark sure you do n
ot fill up before having desserts.  There is so much to choose from an
d everything I had was beautiful and delicious.",
       "Definitely the BEST Buffet in Vegas!!! Tip: Try and get here a
t the Lunch (it's the same as Dinner only costs less)! Brunch/Lunch is
from 12-3:00pm. It was ALL wonderful & Delicious!"],
      dtype=object)
```

Define target variable (for later classification use)

Again, look at perfect (5 stars) and imperfect (1-4 stars) rating

In [29]:
```
df_top_restaurant['favorable'] = (df_top_restaurant['stars'] > 4)
target_top_restaurant = df_top_restaurant['favorable'].values.astype(int)
target_top_restaurant
```

Out[29]:
```
array([0, 1, 0, ..., 0, 0, 1])
```

Check the statistic of the target variable

In [30]:
```
target_top_restaurant.mean()
```

Out[30]:
```
0.3855629465385563
```

In [31]:
```
documents_top_restaurant.shape, target_top_restaurant.shape
```

Out[31]:
```
((4059,), (4059,))
```

Create training dataset and test dataset

In [32]:
```
from sklearn.cross_validation import train_test_split
```

In [33]:
```
# documents_top_restaurant is X, target_top_restaurant is y
# Now split the data to training set and test set
# Now data is smaller
X_train, X_test, y_train, y_test = train_test_split(
    documents_top_restaurant, target_top_restaurant, test_size = 0.4, ran
)
```

In [34]:
```
X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

Out[34]:
```
((2435,), (1624,), (2435,), (1624,))
```

Get NLP representation of the documents

In [35]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
```

In [36]:
```
# Create TfidfVectorizer, and name it vectorizer
vectorizer = TfidfVectorizer(stop_words = 'english', max_features = 1000)
```

In [37]:
```
# Train the model with your training data
vectors_train = vectorizer.fit_transform(X_train).toarray() # toarray avo
```

```
/Users/luoyiting/anaconda/lib/python3.5/site-packages/sklearn/feature_
extraction/text.py:1059: FutureWarning: Conversion of the second argum
ent of issubdtype from `float` to `np.floating` is deprecated. In futu
re, it will be treated as `np.float64 == np.dtype(float).type`.
  if hasattr(X, 'dtype') and np.issubdtype(X.dtype, np.float):
```

In [38]:
```
# Get the vocab of your tfidf
words = vectorizer.get_feature_names()
```

In [39]:
```
# Use the trained model to transform the test data
vectors_test = vectorizer.transform(X_test).toarray()
```

```
/Users/luoyiting/anaconda/lib/python3.5/site-packages/sklearn/feature_
extraction/text.py:1059: FutureWarning: Conversion of the second argum
ent of issubdtype from `float` to `np.floating` is deprecated. In futu
re, it will be treated as `np.float64 == np.dtype(float).type`.
  if hasattr(X, 'dtype') and np.issubdtype(X.dtype, np.float):
```

In [40]:
```
# Use the trained model to transform all the data
vectors_documents_top_restaurant = vectorizer.transform(documents_top_res
```

```
/Users/luoyiting/anaconda/lib/python3.5/site-packages/sklearn/feature_
extraction/text.py:1059: FutureWarning: Conversion of the second argum
ent of issubdtype from `float` to `np.floating` is deprecated. In futu
re, it will be treated as `np.float64 == np.dtype(float).type`.
  if hasattr(X, 'dtype') and np.issubdtype(X.dtype, np.float):
```

Cluster reviews with KMeans

Fit k-means clustering on the training vectors and make predictions on all data

```
In [41]: kmeans = KMeans(n_clusters = 5)
         kmeans.fit(vectors_train)

Out[41]: KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
             n_clusters=5, n_init=10, n_jobs=1, precompute_distances='auto',
             random_state=None, tol=0.0001, verbose=0)
```

Make predictions on all your data

```
In [42]: assigned_cluster = kmeans.predict(vectors_documents_top_restaurant)
```

Inspect the centroids

```
In [43]: print ("cluster centroids:")
         print (kmeans.cluster_centers_.shape)

         cluster centroids:
         (5, 1000)
```

Find the top 10 features for each cluster.

```
In [44]: n_feat = 10
         top_centroids = kmeans.cluster_centers_.argsort()[:, -1:-n_feat:-1]

         print ("top features for each cluster:")
         for num, centroid in enumerate(top_centroids):
             print ("%d: %s" % (num, ",".join(words[i] for i in centroid)))

         top features for each cluster:
         0: line,wait,time,food,long,buffet,hour,got,hours
         1: crab,legs,buffet,king,good,food,prime,oysters,dinner
         2: best,buffet,vegas,food,great,las,buffets,quality,worth
         3: food,good,buffet,place,service,price,great,worth,quality
         4: buffet,seafood,food,section,station,dessert,asian,good,like
```

Print out the rating and review of a random sample of the reviews assigned to each cluster to get a sense of the cluster.

```
In [45]: for i in range(kmeans.n_clusters):
             cluster = np.arange(0, vectors_documents_top_restaurant.shape[0])[ass
             sample_reviews = np.random.choice(cluster, 2, replace = False)
             print ("=" * 10)
             print ("Cluster %d:" % i)
             for review_index in sample_reviews:
                 print("  %s -  " % df.iloc[review_index]['stars']),
                 print("%s" % df.iloc[review_index]['text'])
             print
```

```
==========
Cluster 0:
  4 -
The decor is plain but that's not why you come here. You go to Delmoni
co for the great steaks, sides and yummy desserts. This place is almos
t always five stars for me but this last visit I made then mistake and
deviated from my normal order of filet mignon and did the surf and tur
f special instead. The lobster was good (not sure if I would say amazi
ng since it was slathered in bearnaise sauce) but the steak was only s
o-so; not up to normal Delmonico quality. Covering it in the amazing h
ouse steak sauce did help but I wouldn't recommend ordering the surf a
nd turf.
  3 -
Used my Vegas Rewards points and scored 2 Dinner MGM Grand Buffets it
was great for the price $0...prime rib, crab, shrimp and all the fixin
s
==========
Cluster 1:
  5 -
Food. Was. Amazing.
I tried the Chicken & Waffles with an Apple Compote. It was great!
Bring your appetite!
  5 -
Great Pho!! Had the classic beef pho (number 5), which did not disappo
int with a rich flavorful broth. Would eat here again if I lived in Ve
gas.
==========
Cluster 2:
  3 -
My sister-n-law introduced me to Smashburger when I first moved here a
nd was raving about the burgers. So we went several times, and I did l
ove the burgers. Then, I had a burger somewhere on Decatur that tasted
fishy and that was the last time I ate there a few years ago. I decide
d to give them another try recently. There still not as great as my fi
rst experiences and are just OK. I don't think I'll give them another
try.
  5 -
What seems like a little hole in the wall holds such delicious Filipin
o foods. Came here with a group of 10 and we were able to fit comforta
bly in the restaursnt, but I don't think it can really hold more than
15.

I ordered the Longsilog and everything was delicious, the longanisa, f
ried rice, achara, and egg. I also ordered a melon juice which went ve
ry well with my lunch and tasted very refreshing.

Tip:

Parking is either on the side where the trucks roll in or Park on the
street facing the restaurant.
==========
Cluster 3:
  5 -
```

```
You can be a sucker and wait two hours for bottomless drink options at
some over the top brunch spot on the strip (which I've done), or you c
an take a short drive to this place and risk exploding from eating so 
much deliciousness. We hit this place THREE TIMES this weekend, I just
couldn't say away.  If you're looking for some down to earth, luscious
diner food, make the trip. You won't be disappointed. Short wait times
, friendly staff, and the food is just perf.
  4 -
Yum! Good old fashioned breakfast at a reasonable price. You get a lot
of being for your buck here. I ordered the mini-volcano with the harve
st pancakes. The meal was good, but pancakes were literally falling ap
art. They may need to perfect that recipe a little more. We had good s
ervice too, my coffee never got cold and our waiter was very polite. O
verall it was a good experience and we will be returning.
==========
Cluster 4:
  3 -
It is about $30 for brunch on Saturday without tip.  I would say it is
ok.  I have ate plenty of breakfast buffet in Las Vegas before. I woul
d say go to another place for the bang of the buck.  Its not bad nor g
ood. Only thing I like this past weekend are the apple sausages and Yo
gurt for brunch. Other than that, save your money and try something el
se.  Plenty of better buffets in my opinion that are better.  Great cu
stomer service but just not that good for that price. Waiters and staf
f are friendly and helpful. I give it 3 star. Also if you sign up with
MGM Grand hotel stay, look for promo code. You will get 2 free buffets
per stay.
  2 -
Service average. Atmosphere average. Food....needs significant improve
ment. Well below average.
```

3. Use PCA to reduce dimensionality

Stardardize features

X_train and X_test

standardize features by removing the mean and scaling to the unit varience, to make sure the importance of every feature is equal.

In [46]:
```python
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(vectors_train)
X_test_scaled = scaler.transform(vectors_test)
```

Use PCA to transform data (train and test) and get princial components

In [47]:
```python
from sklearn.decomposition import PCA

n_col = 50
pca = PCA(n_components = n_col)


X_train_pca = pca.fit_transform(X_train_scaled)
X_test_pca = pca.transform(X_test_scaled)
```

In [48]:
```python
X_train_pca.shape, X_test_pca.shape
```

Out[48]:
```
((2435, 50), (1624, 50))
```

See how much (and how much percentage of) variance the principal components explain

In [49]:
```python
print (pca.explained_variance_[:10])
```

```
[8.42908241 6.55077303 4.37295566 4.12875352 3.77257293 3.50508557
 3.40842388 3.25832109 3.22979255 3.09370278]
```

In [50]:
```python
print (pca.explained_variance_ratio_[:10])
```

```
[0.00842908 0.00655077 0.00437296 0.00412875 0.00377257 0.00350509
 0.00340842 0.00325832 0.00322979 0.0030937 ]
```

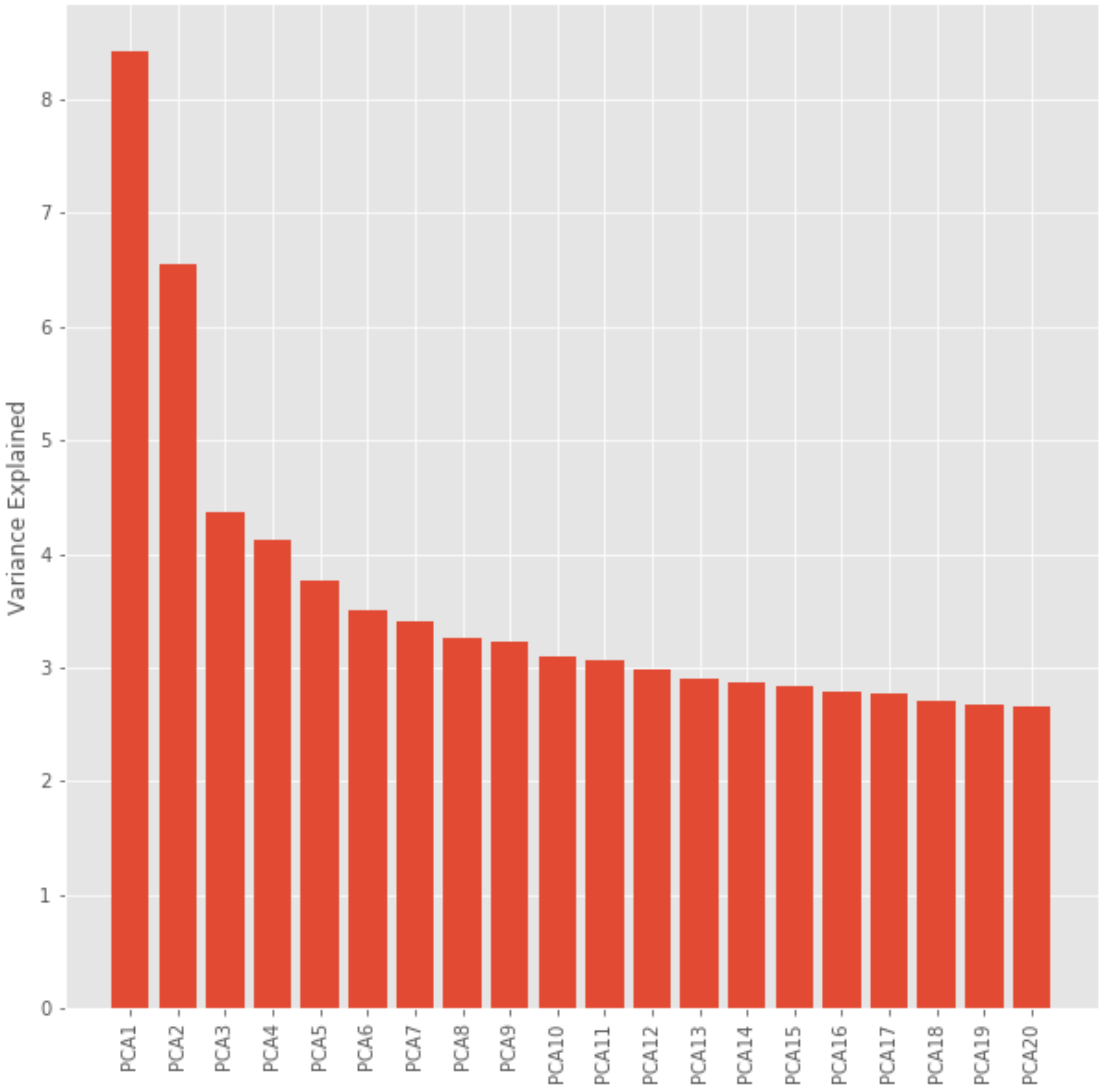
Viz: plot proportion of variance explained with top principal components

For clear display, start with plotting <=20 principal components

```
In [51]: n_col_to_display = 20

         pca_range = np.arange(n_col_to_display) + 1
         pca_names = ["PCA%s" % i for i in pca_range]

         plt.figure(figsize = (10, 10))
         plt.bar(pca_range,
                 pca.explained_variance_[:n_col_to_display],
                 align = 'center')
         xticks = plt.xticks(pca_range,
                             pca_names,
                             rotation = 90)
         plt.ylabel('Variance Explained')
         plt.show()
```



Classifying positive/negative review with PCA preprocessing

Logistic Regression Classifier

Use standardized tf-idf vectors as features (without PCA)

```
In [52]: # Build a Logistic Regression Classifier, train with standardized tf-idf

         from sklearn.linear_model import LogisticRegression

         model_lrc = LogisticRegression()

         model_lrc.fit(X_train_scaled, y_train)
```

```
Out[52]: 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=0.0
         001,
                   verbose=0, warm_start=False)
```

```
In [53]: # Get score for training set
         model_lrc.score(X_train_scaled, y_train)
```

```
Out[53]: 0.9971252566735113
```

```
In [54]: # Get score for test set
         model_lrc.score(X_test_scaled, y_test)
```

```
Out[54]: 0.7112068965517241
```

Use (Stardardized + PCA) tf-idf vectors as features

In [55]:
```
# Build a Logistic Regression Classifier, train with PCA tranformed X

from sklearn.linear_model import LogisticRegression

model_lrc = LogisticRegression()

model_lrc.fit(X_train_pca, y_train)
```

Out[55]:
```
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=0.0
001,
          verbose=0, warm_start=False)
```

In [56]:
```
# Get score for training set
model_lrc.score(X_train_pca, y_train)
```

Out[56]:
```
0.7860369609856263
```

In [57]:
```
# Get score for test set, REMEMBER to use PCA-transformed X!
model_lrc.score(X_test_pca, y_test)
```

Out[57]:
```
0.7869458128078818
```

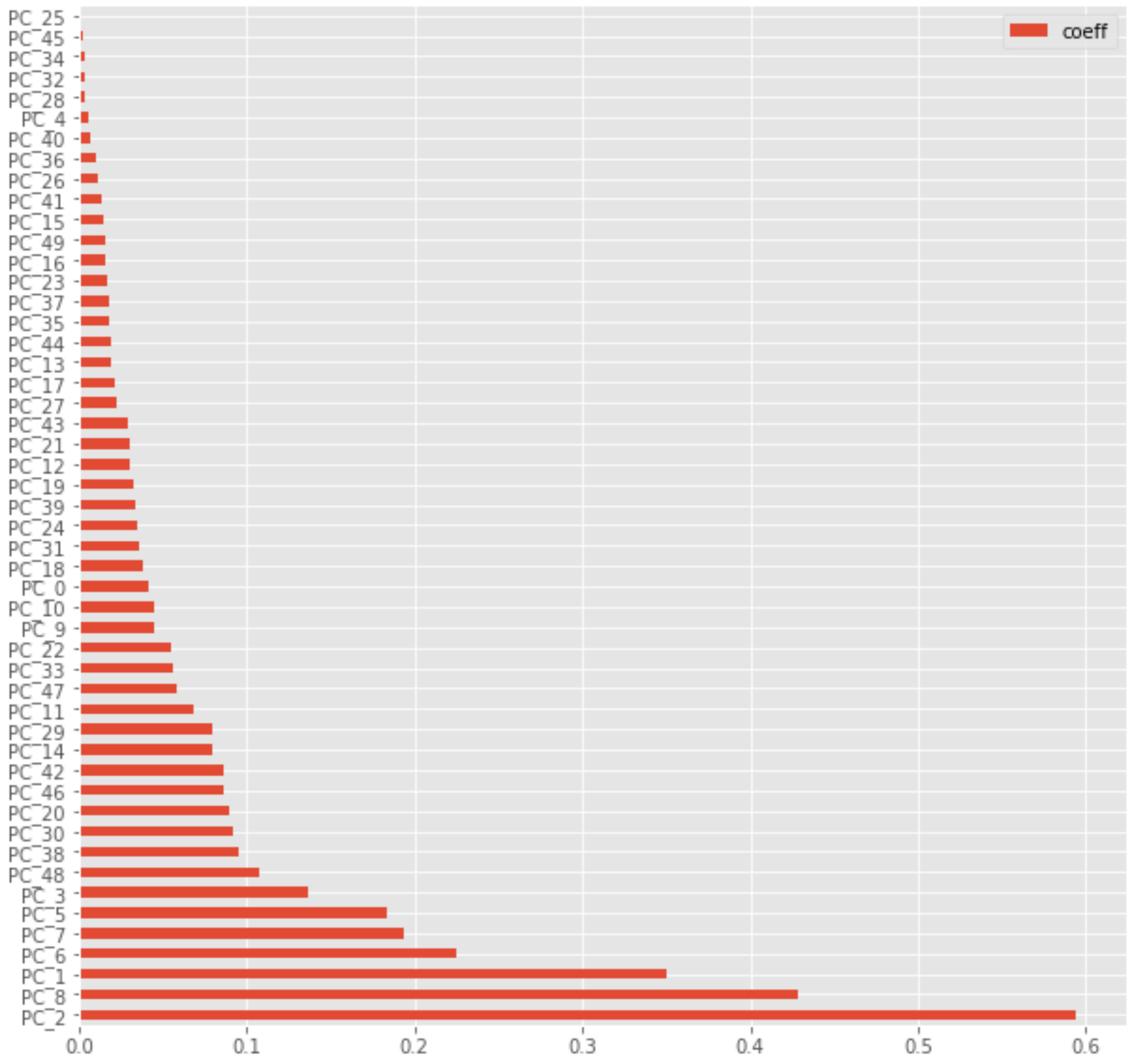
Thus, in this case, PCA helps us to avoid over-fitting. However, the interpretation of standarization + PCA is worse than standarization only.

Plot the coefficients against principal components

```
In [58]: pca_range = np.arange(pca.n_components_)
         pca_names = ['PC_%s' % i for i in pca_range]

         df_coeffs = pd.DataFrame(list(zip(pca_names, abs(model_lrc.coef_.flatten(
         df_coeffs.columns = ['PCs', 'coeff']

         ax = df_coeffs.plot.barh(figsize = (10, 10))
         t = np.arange(pca.n_components_)
         ax.set_yticks(t)
         ax.set_yticklabels(df_coeffs['PCs'])
         plt.show()
```



Random Forest Classifier

Use standardized tf-idf vectors as features

In [59]:
```
# Build a Random Forest Classifier

from sklearn.ensemble import RandomForestClassifier

model_rfc = RandomForestClassifier (max_depth = None,
                                    n_estimators = 40,
                                    min_samples_leaf = 3,
                                    random_state = 42)
model_rfc.fit(X_train_scaled, y_train)
```

Out[59]:
```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='g
ini',
            max_depth=None, max_features='auto', max_leaf_nodes=None,
            min_impurity_split=1e-07, min_samples_leaf=3,
            min_samples_split=2, min_weight_fraction_leaf=0.0,
            n_estimators=40, n_jobs=1, oob_score=False, random_state=4
2,
            verbose=0, warm_start=False)
```

In [60]:
```
# Get score for training set
model_rfc.score(X_train_scaled, y_train)
```

Out[60]:
```
0.9535934291581109
```

In [61]:
```
# Get score for test set
model_rfc.score(X_test_scaled, y_test)
```

Out[61]:
```
0.7450738916256158
```

Use (Stardardized + PCA) tf-idf vectors as features

In [62]:
```
# Build a Random Forest Classifier

from sklearn.ensemble import RandomForestClassifier

model_rfc = RandomForestClassifier(max_depth = None,
                                   n_estimators = 40,
                                   min_samples_leaf = 3,
                                   random_state = 42)
model_rfc.fit(X_train_pca, y_train)
```

Out[62]:
```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='g
ini',
            max_depth=None, max_features='auto', max_leaf_nodes=None,
            min_impurity_split=1e-07, min_samples_leaf=3,
            min_samples_split=2, min_weight_fraction_leaf=0.0,
            n_estimators=40, n_jobs=1, oob_score=False, random_state=4
2,
            verbose=0, warm_start=False)
```

```
In [63]: # Get score for training set
         model_rfc.score(X_train_pca, y_train)

Out[63]: 0.988911704312115
```

```
In [64]: # Get score for test set, REMEMBER to use PCA-transformed X!
         model_rfc.score(X_test_pca, y_test)

Out[64]: 0.7469211822660099
```

Generally, random forest model is not easy to be over-fitted because this model decrease variance. In this case, the above result shows that random forest is over-fitted, we can guess it may because of large amount of noise in the data.

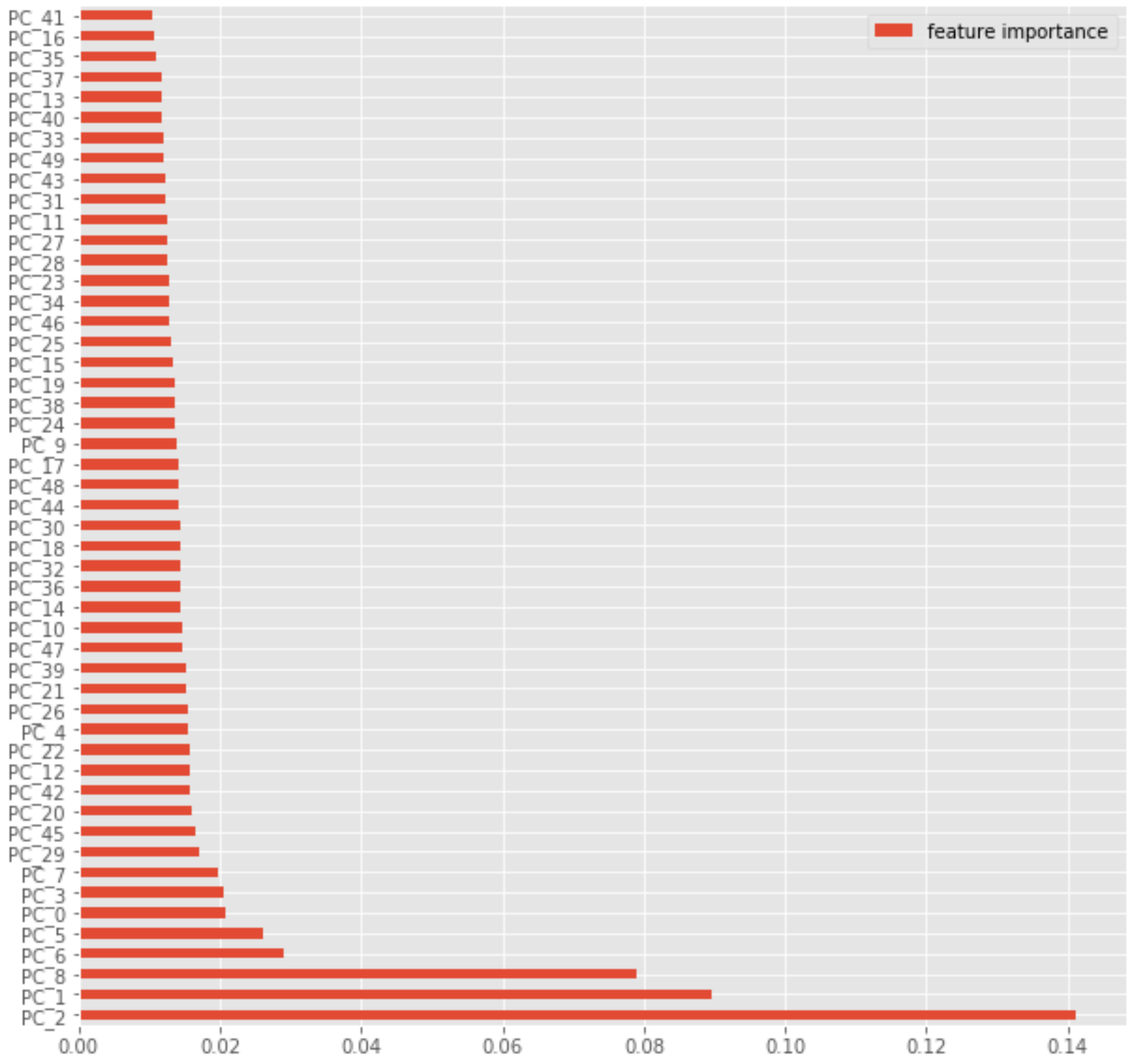
PCA doesn't have much effect here.

Plot the feature importances against principal components

```
In [66]: pca_range = np.arange(pca.n_components_)
         pca_names = ['PC_%s' % i for i in pca_range]

         df_coeffs = pd.DataFrame(list(zip(pca_names, model_rfc.feature_importance
         df_coeffs.columns = ['PCs', 'feature importance']

         ax = df_coeffs.plot.barh(figsize = (10, 10))
         t = np.arange(pca.n_components_)
         ax.set_yticks(t)
         ax.set_yticklabels(df_coeffs['PCs'])
         plt.show()
```



```
In [ ]:
```