In [52]:

```
import sys
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
sns.set()
```

In [2]:

```
df=pd.read_csv("G:/研究生学习资料/Illinois Courses/Fall 2019/IE 598 Machine Learning/assignment/HW3/
df.head()
```

Out[2]:

	CUSIP	Ticker	Issue Date	Maturity	1st Call Date	Moodys	S_and_P	Fitch	Bloomberg Composite Rating
0	000324AA1	FLECIN	7/1/2014	7/1/2019	10/23/2017	Nan	Nan	Nan	Na
1	00080QAB1	RBS	3/15/2004	6/4/2018	Nan	Ba1	BB+	BBB	BB
2	00081TAD0	ACCO	5/14/2010	3/15/2015	Nan	WR	NR	BB+	N
3	00081TAH1	ACCO	6/17/2013	4/30/2020	Nan	WR	NR	WD	N
4	00081TAJ7	ACCO	12/22/2016	12/15/2024	12/15/2019	B1	BB-	BB	BB

5 rows × 37 columns

In [11]:

```
print("Number of Rows of Data = "+ str(df.shape[0]))
print("Number of Columns of Data = "+ str(df.shape[1]))
print("CSV size: "+str(df.shape))
```

```
Number of Rows of Data = 2721
Number of Columns of Data = 37
CSV size: (2721, 37)
```

In [31]:

```
print (df.iloc[0,:])
#Reference: https://zhuanlan.zhihu.com/p/31360526
```

```
CUSIP                           000324AA1
Ticker                             FLECIN
Issue Date                       7/1/2014
Maturity                         7/1/2019
1st Call Date                  10/23/2017
Moodys                                Nan
S_and_P                               Nan
Fitch                                 Nan
Bloomberg Composite Rating            Nan
Coupon                                 12
Issued Amount                    4.05e+08
Maturity Type                    CALLABLE
Coupon Type                   PAY-IN-KIND
Maturity At Issue months            60.87
Industry                      Real Estate
LiquidityScore                    10.8914
Months in JNK                         Nan
Months in HYG                         Nan
Months in Both                        Nan
IN_ETF                                 No
LIQ SCORE                        0.108914
n_trades                              301
volume_trades                 2.64004e+08
total_median_size                   1e+06
total_mean_size                    877089
n_days_trade                          128
days_diff_max                        1132
percent_intra_dealer           0.00664452
percent_uncapped                 0.292359
bond_type                               5
Client_Trade_Percentage          0.521595
weekly_mean_volume            3.10593e+06
weekly_median_volume                2e+06
weekly_max_volume               1.898e+07
weekly_min_volume                   60000
weekly_mean_ntrades               3.54118
weekly_median_ntrades                   1
Name: 0, dtype: object
```

In [32]:

```
print(df.dtypes)
```

```
CUSIP                          object
Ticker                         object
Issue Date                     object
Maturity                       object
1st Call Date                  object
Moodys                         object
S_and_P                        object
Fitch                          object
Bloomberg Composite Rating     object
Coupon                        float64
Issued Amount                 float64
Maturity Type                  object
Coupon Type                    object
Maturity At Issue months      float64
Industry                       object
LiquidityScore                float64
Months in JNK                  object
Months in HYG                  object
Months in Both                 object
IN_ETF                         object
LIQ SCORE                     float64
n_trades                        int64
volume_trades                 float64
total_median_size             float64
total_mean_size               float64
n_days_trade                    int64
days_diff_max                   int64
percent_intra_dealer          float64
percent_uncapped              float64
bond_type                       int64
Client_Trade_Percentage       float64
weekly_mean_volume            float64
weekly_median_volume          float64
weekly_max_volume             float64
weekly_min_volume             float64
weekly_mean_ntrades           float64
weekly_median_ntrades           int64
dtype: object
```

In [34]:

```
# descriptive statistics for the numeric variables
percentiles = np.array([2.5, 25, 50, 75, 97.5])
df.describe()
#https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.describe.html
```

Out[34]:

	Coupon	Issued Amount	Maturity At Issue months	LiquidityScore	LIQ SCORE	n_trades	volu
count	2721.000000	2.721000e+03	2721.000000	2721.000000	2721.000000	2721.000000	2.7
mean	10.307872	8.299295e+08	113.968997	18.218230	0.182182	2700.696435	7.2
std	63.051382	5.802790e+08	101.893176	7.872071	0.078721	5572.262205	1.0
min	0.000000	3.700000e+08	11.930000	4.388758	0.043888	1.000000	7.0
25%	5.000000	5.000000e+08	65.170000	12.738630	0.127386	116.000000	6.1
50%	6.250000	6.500000e+08	97.370000	16.538471	0.165385	674.000000	3.4
75%	7.750000	1.000000e+09	121.770000	22.120108	0.221201	2467.000000	9.3
max	999.000000	7.364026e+09	1217.570000	54.673908	0.546739	57935.000000	8.9

8 rows × 21 columns

In [56]:

```
# unique categories in each categorical attribute
BCR=df['Bloomberg Composite Rating']
Categories=set(BCR)
sys.stdout.write("Unique Label Values \n")
print(Categories)
sys.stdout.write(" \n")
```

```
Unique Label Values
{'AA-', 'CC+', 'A+', 'BBB+', 'DDD', 'CCC-', 'CC-', 'BBB-', 'CCC', 'A-', 'BB-', 'BB
B', 'AAA', 'AA+', 'Nan', 'NR', 'BB', 'B', 'C+', 'DD+', 'B+', 'CC', 'BB+', 'B-', 'CCC
+', 'A', 'AA', 'C'}
```

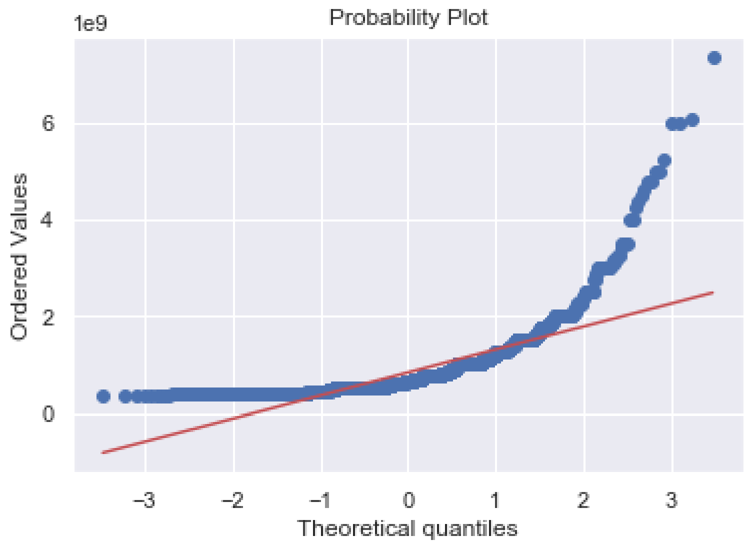
In [57]:

```
sys.stdout.write("\nCounts for Each Value of Categorical Label \n")
catCount={}
for elt in BCR:
    if elt in catCount:
        catCount[elt] += 1
    else:
        catCount[elt]=1
print(catCount)
```

```
Counts for Each Value of Categorical Label
{'Nan': 41, 'BB+': 258, 'NR': 1136, 'BB-': 196, 'BB': 179, 'AA-': 63, 'A': 22, 'B-':
124, 'B+': 150, 'B': 116, 'A+': 26, 'CCC+': 70, 'CCC': 54, 'BBB-': 163, 'BBB': 33,
'CCC-': 16, 'CC': 6, 'CC+': 16, 'AA': 7, 'AAA': 12, 'A-': 8, 'DDD': 2, 'BBB+': 11,
'DD+': 2, 'C': 3, 'C+': 5, 'CC-': 1, 'AA+': 1}
```

In [66]:

```
import scipy.stats as stats
fig=plt.figure()
ax=fig.add_subplot(111)
stats.probplot(df['Issued Amount'], dist="norm",plot=ax)
plt.show()
```



In [67]:

```
from pandas import DataFrame
print(df.head())
print(df.tail())
summary = df.describe()
print(summary)
```

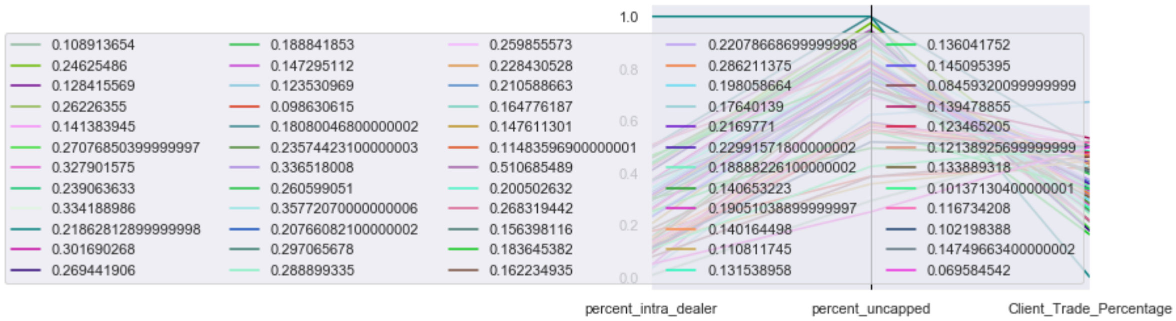
```
       CUSIP  Ticker  Issue Date    Maturity 1st Call Date Moodys S_and_P  \
0  000324AA1  FLECIN    7/1/2014    7/1/2019    10/23/2017    Nan     Nan
1  00080QAB1     RBS   3/15/2004    6/4/2018           Nan    Ba1     BB+
2  00081TAD0    ACCO   5/14/2010   3/15/2015           Nan     WR      NR
3  00081TAH1    ACCO   6/17/2013   4/30/2020           Nan     WR      NR
4  00081TAJ7    ACCO  12/22/2016  12/15/2024    12/15/2019     B1     BB-

  Fitch Bloomberg Composite Rating  Coupon  ...   percent_intra_dealer  \
0   Nan                        Nan   12.00  ...               0.006645
1   BBB                        BB+    4.65  ...               0.425018
2   BB+                         NR   10.63  ...               0.115207
3    WD                         NR    6.75  ...               0.426332
4    BB                        BB-    5.25  ...               0.157216

  percent_uncapped bond_type  Client_Trade_Percentage weekly_mean_volume  \
0         0.292359         5                 0.521595        3105926.765
1         0.974071         2                 0.337071        1721696.774
2         0.594470         5                 0.467742        4200313.433
3         0.892462         3                 0.212864        6321559.783
```

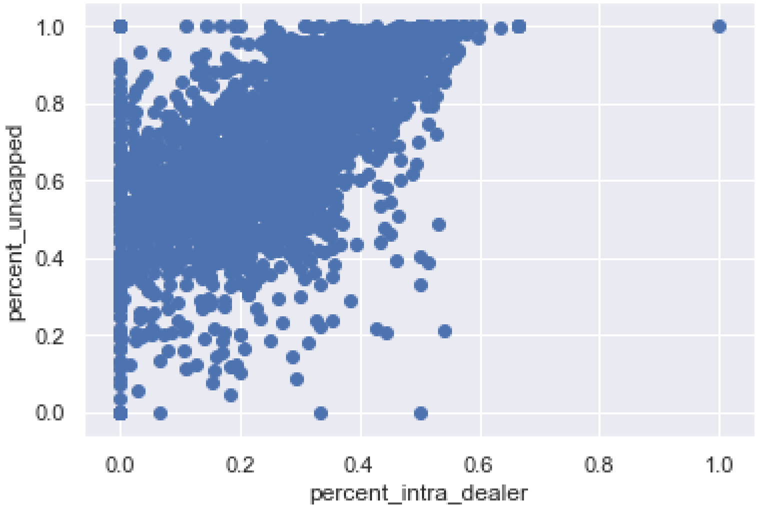
In [96]:

```
from pandas.plotting import parallel_coordinates
tmp = df[['LIQ SCORE','percent_intra_dealer','percent_uncapped','Client_Trade_Percentage']][0:60]
parallel_coordinates(tmp,'LIQ SCORE')
plt.legend(loc='best',ncol=5)
plt.show()
```



In [98]:

```
plt.scatter(df['percent_intra_dealer'],df['percent_uncapped'])
plt.xlabel("percent_intra_dealer")
plt.ylabel(("percent_uncapped"))
plt.show()
```

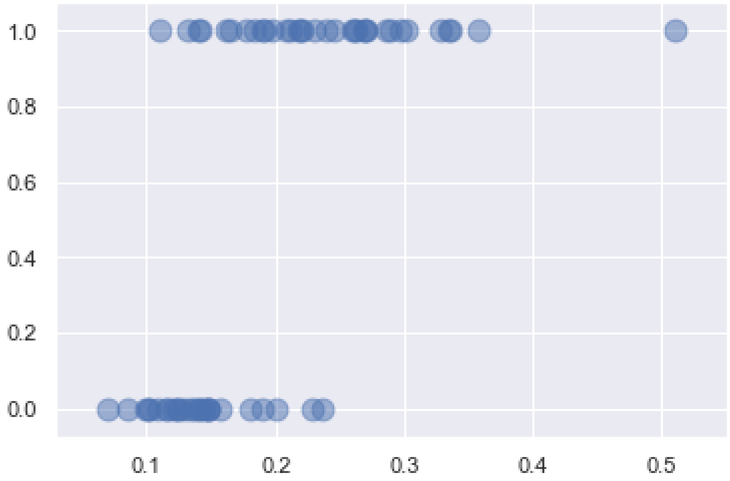


In [117]:

```
target=[]
for i in range(60):
    if df['IN_ETF'][i] == "Yes":
        target.append(1.0)
    else:
        target.append(0.0)
tmp=np.array(target)
plt.scatter(df['LIQ SCORE'][:60], target, alpha=0.5, s=120)
```

Out[117]:

```
<matplotlib.collections.PathCollection at 0x2095ba94128>
```



In [120]:

```
sys.stdout.write("Correlation between attribute V(n_trades) and W(volume_trades) \n")
tmp=np.corrcoef(df['n_trades'], df['volume_trades'])[0, 1]
print(tmp)
```

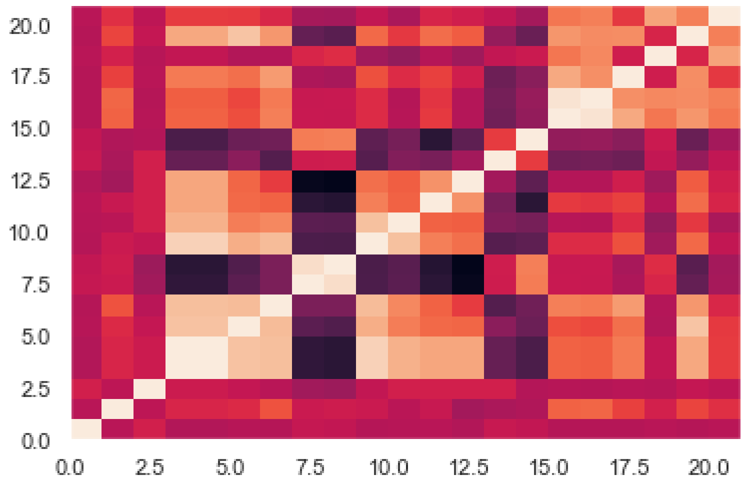
```
Correlation between attribute V(n_trades) and W(volume_trades)
0.7693223728724927
```

In [121]:

```
from pandas import DataFrame
corMat = DataFrame(df.corr())
```

In [122]:

```
plt.pcolor(corMat)
plt.show()
```



In [123]:

```
print("My name is {Zihan Chen}")
print("My NetID is: {zihanc7}")
print("I hereby certify that I have read the University policy on Academic Integrity and that I am r
```

```
My name is {Zihan Chen}
My NetID is: {zihanc7}
I hereby certify that I have read the University policy on Academic Integrity and th
at I am not in violation.
```

In [ ]: