Objective:

1. Perform EDA on HY_Universe_corporate bond

In [1]:
```
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as  np
import pandas as  pd
import os
import math
import re
pd.options.display.max_columns=40
import seaborn as sns
```

In [2]:
```
### Readin the data
path="C:\\Users\\fbaharkoush\\IE 598 Machine Learning\\Homework\\HW 3\\"
df_bond=pd.read_csv(path+"HY_Universe_corporate bond.csv",)
```

2.1

In [3]:
```
### Number of Rows and Col
print("Number of rows of Data=",df_bond.shape[0])
print("Number of columns of Data=",df_bond.shape[1])
```

```
Number of rows of Data= 2721
Number of columns of Data= 37
```

Fillna Na of numericla cols

In [4]:
```
list_of_numerical_col_with_nan=["Months in JNK","Months in HYG","Months in Both"]
df_bond[list_of_numerical_col_with_nan]=df_bond[list_of_numerical_col_with_nan].replace("Nan",0).astype(float)
```

2.2

In [5]:
```
### Identify the types of vallues in columns
```

In [6]:
```
### Initiate List to count data Types
list_of_number_of_str=[]
list_of_number_of_float=[]
list_of_number_of_int=[]
list_of_columns=[]
for i in list(df_bond.columns):
    number_of_str=df_bond[i].apply(lambda x: type(x)==str).sum()
    number_of_float=df_bond[i].apply(lambda x: type(x)==float).sum()
    number_of_int=df_bond[i].apply(lambda x: type(x)==int).sum()
    list_of_number_of_str.append(number_of_str)
    list_of_number_of_float.append(number_of_float)
    list_of_number_of_int.append(number_of_int)
    list_of_columns.append(i)
```

In [7]:
```
df_bond_dtype_count=pd.DataFrame({"Columns":list_of_columns,
              "number_of_str":list_of_number_of_str,
              "number_of_float":list_of_number_of_float,
               "number_of_int":list_of_number_of_int})
### Count Other Data Type
df_bond_dtype_count["Other"]=df_bond_dtype_count.sum(axis=1)-df_bond.shape[0]
```

In [8]:
```
df_bond_dtype_count.head()
```

Out[8]:

	Columns	number_of_str	number_of_float	number_of_int	Other
0	CUSIP	2721	0	0	0
1	Ticker	2721	0	0	0
2	Issue Date	2721	0	0	0
3	Maturity	2721	0	0	0
4	1st Call Date	2721	0	0	0

In [9]:
```
list_of_numerical_col=list(df_bond_dtype_count[df_bond_dtype_count["number_of_str"]==0]["Columns"])
list_of_categorical_col=list(df_bond_dtype_count[df_bond_dtype_count["number_of_str"]!=0]["Columns"])
```

2.3

In [10]:
```
### df.describe() provide the summary of all the numberical columns
df_bond[list_of_numerical_col].describe()
```

Out[10]:

	Coupon	Issued Amount	Maturity At Issue months	LiquidityScore	Months in JNK	Months in HYG	Months in Both	LIQ SCORE	n_trades	volume_trades	total_median_s
count	2721.000000	2.721000e+03	2721.000000	2721.000000	2721.000000	2721.000000	2721.000000	2721.000000	2721.000000	2.721000e+03	2.721000e
mean	10.307872	8.299295e+08	113.968997	18.218230	7.703785	10.496141	5.724366	0.182182	2700.696435	7.222372e+08	5.361476e
std	63.051382	5.802790e+08	101.893176	7.872071	13.906823	16.830778	12.267923	0.078721	5572.262205	1.027825e+09	4.193546e
min	0.000000	3.700000e+08	11.930000	4.388758	0.000000	0.000000	0.000000	0.043888	1.000000	7.000000e+03	4.000000e
25%	5.000000	5.000000e+08	65.170000	12.738630	0.000000	0.000000	0.000000	0.127386	116.000000	6.189000e+07	7.500000e
50%	6.250000	6.500000e+08	97.370000	16.538471	0.000000	0.000000	0.000000	0.165385	674.000000	3.480000e+08	5.000000e
75%	7.750000	1.000000e+09	121.770000	22.120108	10.000000	16.000000	3.000000	0.221201	2467.000000	9.328420e+08	1.000000e
max	999.000000	7.364026e+09	1217.570000	54.673908	64.000000	67.000000	63.000000	0.546739	57935.000000	8.979960e+09	3.400000e

In [11]:
```
df_bond_categorical=df_bond[list_of_categorical_col].drop(["Issue Date","Maturity","1st Call Date","CUSIP"],axis=1)
```

In [12]:
```
df_bond_categorical.head()
```

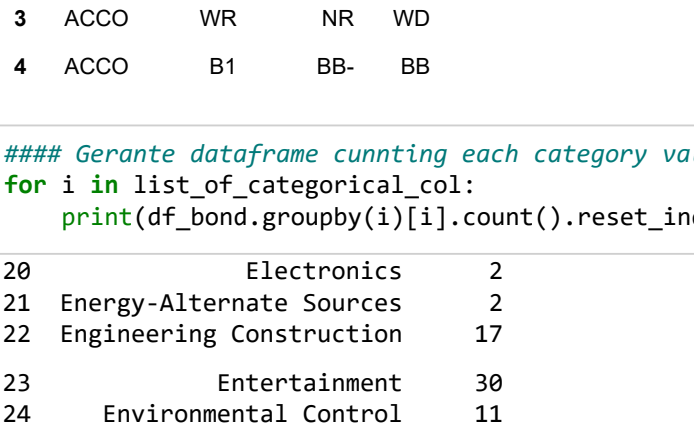
Out[12]:

	Ticker	Moodys	S_and_P	Fitch	Bloomberg Composite Rating	Maturity Type	Coupon Type	Industry	IN_ETF
0	FLECIN	Nan	Nan	Nan	Nan	CALLABLE	PAY-IN-KIND	Real Estate	No
1	RBS	Ba1	BB+	BBB	BB+	AT MATURITY	FIXED	Banks	Yes
2	ACCO	WR	NR	BB+	NR	CALLABLE	FIXED	Household Products/Wares	No
3	ACCO	WR	NR	WD	NR	CALLABLE	FIXED	Household Products/Wares	Yes
4	ACCO	B1	BB-	BB	BB-	CALLABLE	FIXED	Household Products/Wares	No

In [13]:
```
#### Gerante dataframe cunnting each category value in each col
for i in list_of_categorical_col:
    print(df_bond.groupby(i)[i].count().reset_index(name="Count"))
```

```
20              Electronics      2
21  Energy-Alternate Sources      2
22  Engineering Construction     17
23             Entertainment     30
24     Environmental Control     11
25                      Food     46
26              Food Service     10
27     Forest Products Paper     22
28                       Gas     20
29        Hand/Machine Tools      3
..                       ...    ...
39      Investment Companies      3
40                Iron/Steel     57
41              Leisure Time      9
42                   Lodging     46
43   Machinery-Constr Mining      5
44     Machinery-Diversified     22
45                     Media    131
46  Metal Fabricate/Hardware      6
```

In [14]:
```
for i in list_of_numerical_col:
    plt.boxplot(df_bond[i])
    plt.xlabel(i)
    plt.show()
```

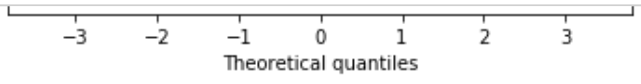


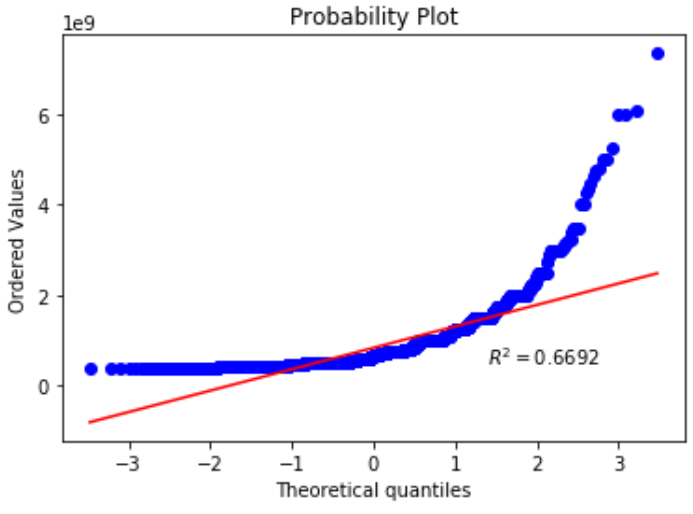
In [ ]:
```
```

2.4

In [15]:
```
import pylab
import scipy.stats as stats
```

In [16]:
```
for i in list_of_numerical_col:
    measurements =df_bond[i]
    stats.probplot(measurements, dist="norm", plot=pylab,rvalue=True)
    pylab.show()
```





2.5

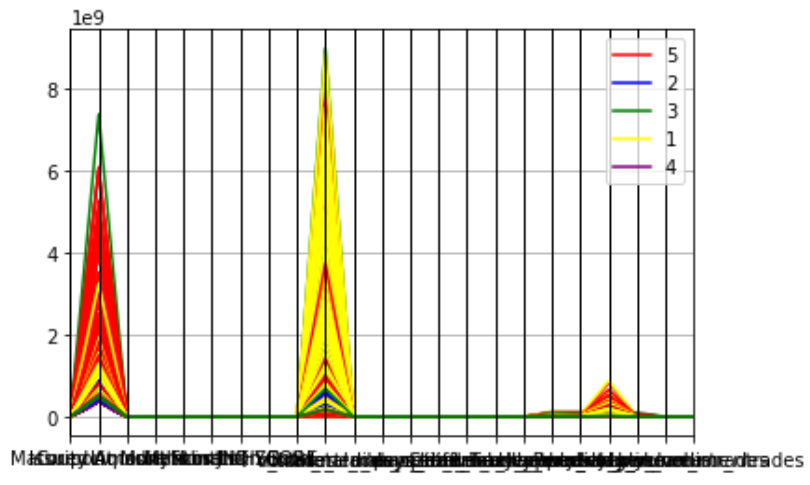
In [17]:
```
df_bond.describe()
```

Out[17]:

	Coupon	Issued Amount	Maturity At Issue months	LiquidityScore	Months in JNK	Months in HYG	Months in Both	LIQ SCORE	n_trades	volume_trades	total_median_s
count	2721.000000	2.721000e+03	2721.000000	2721.000000	2721.000000	2721.000000	2721.000000	2721.000000	2721.000000	2.721000e+03	2.721000e
mean	10.307872	8.299295e+08	113.968997	18.218230	7.703785	10.496141	5.724366	0.182182	2700.696435	7.222372e+08	5.361476e
std	63.051382	5.802790e+08	101.893176	7.872071	13.906823	16.830778	12.267923	0.078721	5572.262205	1.027825e+09	4.193546e
min	0.000000	3.700000e+08	11.930000	4.388758	0.000000	0.000000	0.000000	0.043888	1.000000	7.000000e+03	4.000000e
25%	5.000000	5.000000e+08	65.170000	12.738630	0.000000	0.000000	0.000000	0.127386	116.000000	6.189000e+07	7.500000e
50%	6.250000	6.500000e+08	97.370000	16.538471	0.000000	0.000000	0.000000	0.165385	674.000000	3.480000e+08	5.000000e
75%	7.750000	1.000000e+09	121.770000	22.120108	10.000000	16.000000	3.000000	0.221201	2467.000000	9.328420e+08	1.000000e
max	999.000000	7.364026e+09	1217.570000	54.673908	64.000000	67.000000	63.000000	0.546739	57935.000000	8.979960e+09	3.400000e

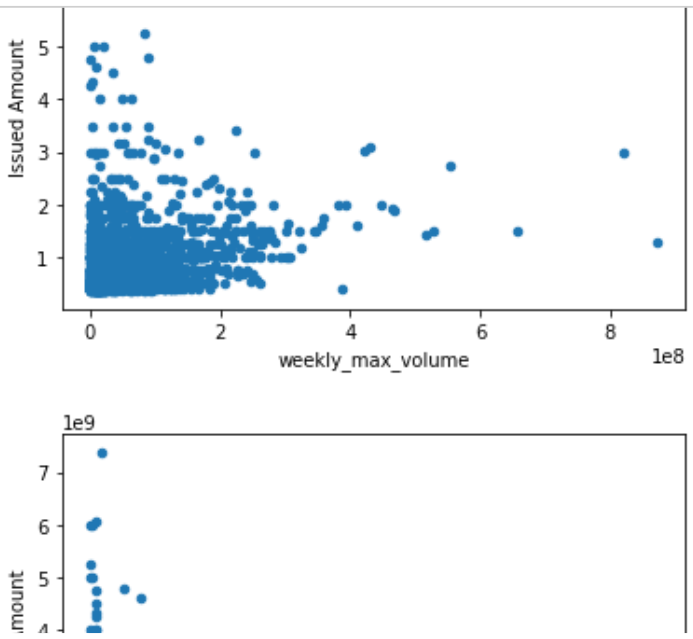
2.6

In [18]:
```
### parallel coordinates plot for Bonds Type 1,2,3,4
### obviously it is not a good graph since there are many numerical values as feaers for Bond Types
pd.plotting.parallel_coordinates(df_bond[list_of_numerical_col],"bond_type",
                                 color=["red","blue","green","yellow","purple"])
plt.show()
```



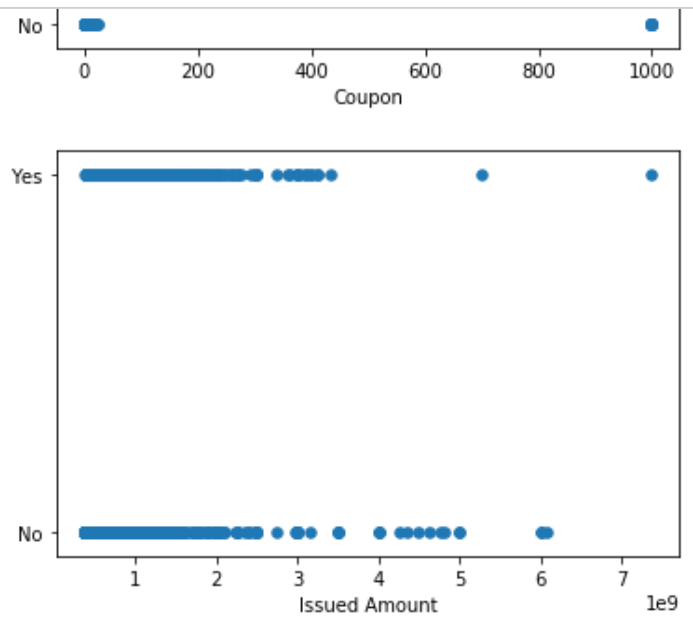
2.7

In [19]:
```
for i in list_of_numerical_col:
    df_bond.plot.scatter(x=i, y='Issued Amount')
    plt.show()
```



2.8

In [20]:
```
### Scatter Plot for Numerical Values vs IN_ETF
for i in list_of_numerical_col:
    plt.scatter(df_bond[i], df_bond["IN_ETF"], s=30)
    plt.xlabel(i)
    plt.show()
```



2.9

I used https://stackoverflow.com/questions/29432629/plot-correlation-matrix-using-pandas (https://stackoverflow.com/questions/29432629/plot-correlation-matrix-using-pandas) as guidance

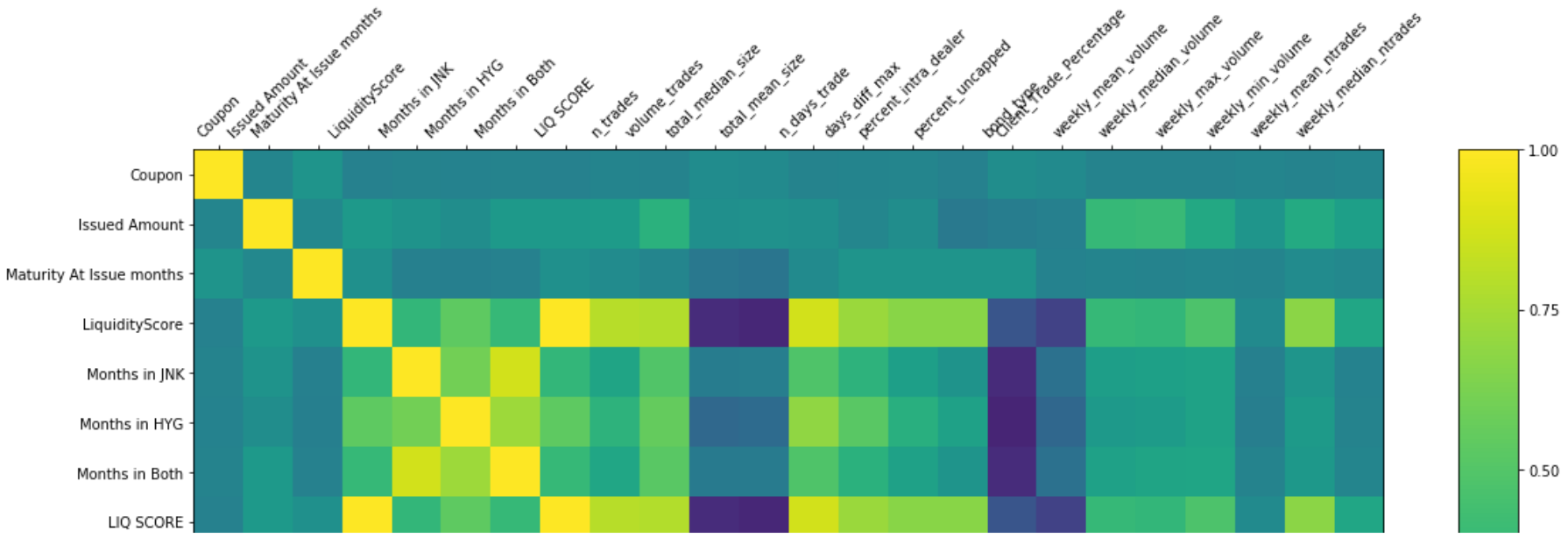
In [21]:
```
### Correlation Matrix
corr = df_bond[list_of_numerical_col].corr()
corr.style.background_gradient(cmap='coolwarm')
```

n_days_trade	-0.0283363	0.068113	0.0295302	0.87304	0.485822	0.687806	0.486494	0.87304	0.70431	0.772564
days_diff_max	-0.0250892	-0.00809704	0.103178	0.71728	0.344136	0.522358	0.326563	0.71728	0.497633	0.540932
percent_intra_dealer	-0.0143163	0.0526173	0.104127	0.671903	0.186363	0.311058	0.201814	0.671903	0.415695	0.387555
percent_uncapped	-0.0458972	-0.112369	0.100168	0.666321	0.0933964	0.201064	0.0996741	0.666321	0.39688	0.241814
bond_type	0.0518559	-0.0707143	0.10299	-0.368492	-0.632745	-0.672686	-0.625617	-0.368492	-0.208283	-0.452584
Client_Trade_Percentage	0.0291248	-0.0495129	-0.0401862	-0.496127	-0.168572	-0.243049	-0.164876	-0.496127	-0.348408	-0.327922
weekly_mean_volume	-0.0277242	0.38205	-0.0230016	0.385978	0.179907	0.146767	0.206252	0.385978	0.309053	0.503159
weekly_median_volume	-0.0285842	0.396947	-0.0328677	0.371213	0.194785	0.16186	0.223336	0.371213	0.285998	0.479018
weekly_max_volume	-0.0263617	0.261469	-0.0171373	0.481142	0.212305	0.208108	0.235184	0.481142	0.432955	0.616802

2.10

I used https://stackoverflow.com/questions/29432629/plot-correlation-matrix-using-pandas (https://stackoverflow.com/questions/29432629/plot-correlation-matrix-using-pandas) as guidance

In [22]:
```
### Heat Map for Numberical Values
f = plt.figure(figsize=(19, 15))
plt.matshow(df_bond[list_of_numerical_col].corr(), fignum=f.number)
plt.xticks(range(df_bond[list_of_numerical_col].shape[1]), df_bond[list_of_numerical_col].columns, fontsize=10, rotation=45)
plt.yticks(range(df_bond[list_of_numerical_col].shape[1]), df_bond[list_of_numerical_col].columns, fontsize=10)
cb = plt.colorbar()
cb.ax.tick_params(labelsize=10)
```



In [23]:
```
print("My name is Farbod Baharkoush")
print("My NetID is: fbahar2")
print("I hereby certify that I have read the University policy on Academic Integrity and that I am not in violation.")
```

```
My name is Farbod Baharkoush
My NetID is: fbahar2
I hereby certify that I have read the University policy on Academic Integrity and that I am not in violation.
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: