Experiment 4

Aim: Compare and analyse Mutual Funds (India) dataset and build a model for recommendation of good schemes and understanding factors affecting scheme performance over the short, medium and long terms.

Theory:

Mutual fund: A mutual fund is an investment vehicle that is made up of a pool of funds collected from many investors for the purpose of investing in securities such as stocks, bonds, money market instruments and similar assets.

The power of mutual funds lies in that they are managed by "fund managers", who invest the fund's capital and attempt to produce capital gains and income for the fund's investors. This enables small investors to invest in professionally managed, diversified portfolios of equities, bonds and other securities, which would otherwise be very hard to create with limited knowledge and a small amount of capital.

There are various factors to consider while making an **investment decision:**

- **Desired Income:** A regular current income or long-term capital gains or tax benefits, etc.?
- **Risk Appetite:** A low-risk, low-gain conservative portfolio or high-risk, high gain portfolio in volatile categories?
- **Time Horizon:** Liquidity concerns in the short, medium and long terms
- **Fund type:** Capital appreciation in equity funds or mixed investment in stocks and bonds using balanced funds?
- **Fund category:** Diversified or Narrow? Blue-chip or Energy? etc.
- Size of the fund: Assets managed by the fund
- **Historical Returns**
- **Benchmarks** and benchmark performance ...

Only less than 10% of Indian households invest in mutual fund schemes, despite them being fairly well regulated by Association of Mutual Funds in India and being managed by professional fund managers due to lack of information on how mutual funds work. This motivates for undertaking this experiment.

Problem Statement

The aim of this experiment is to perform an analysis of mutual fund schemes in India to recommend good investment options. Also to educate users about the factors and the degree to which these factors affect the mutual fund performance in different time horizons ranging from 1 month to 5 years.

Therefore, pursuing such an analysis would be a huge avenue for business development, growing and sustaining customers by offering the unique value add in the form of educating them, and a major incentive to the business itself in terms of developing in-house understanding of mutual fund performance dynamics.

Data Processing Pipeline

Sources of Data

The data of mutual fund families and individual schemes operating in India is taken from two disparate data sources:

1. Association of Mutual Funds in India (or AMFI) [4] - the association of SEBI (Securities and Exchange Board of India) registered mutual funds in India of all the registered Asset Management Companies. [5]
2. MoneyControl.com [6] - largest online financial platform in India [7]

Pipeline

Pre-processing:

As the data from either of the sources is not available as a structured dataset available for download, web-scrapers to scrape latest net asset value (NAV) data from AMFI [5] which is updated daily and detailed financial information about the fund as well as returns history from MoneyControl.com is written.

The source data need to be collected and stored as CSVs. The fund family and fund schemes data for analysis lies in a list of fund schemes, each of which is represented as a dictionary (key-value store) of various attributes of the fund scheme. The data has to be cleaned and transformed extensively to be of any practical use. Some of the data cleaning, preprocessing and transformation steps that were performed are listed:

1. While scraping AMFI data, patterns have been identified in data to parse the structure. By going over the contents of the file, observe that

    a. The first line represents the titles of columns in a ;-delimited file

    b. There are blank lines that have to be ignored

    c. There are lines with only text, and no ;-delimited values, which may represent either mutual fund scheme's type or the name of a fund family

    - If a single line of text is encountered before a line containing ;-delimited values it is to be interpreted as the fund family name for all funds until next such line is encountered

- If two lines of text are encountered before a line containing ;-delimited values, then the first line is to be interpreted as scheme type and the second line as the fund family name for all funds until next such line is encountered d. Create extra fields for better representation of data to join with other dataset - namely, scheme classification, type, category, fund family, ID and a short name. All of these are derived from composite data appearing in a single field.

2. While scraping MoneyControl data:

   a. Collect everything as unicode strings, and fill missing values ('NA', 'N.A.', '--', '-', None) with unicode text 'None' instead of the None keyword for the sake of consistency and ease of processing later

   b. Implemented a method encode_risk() to encode risk into a numeric value on a scale from 1 to 5. The higher the risk, the lower the score.

   c. Implemented a method to_numeric() to convert all categorical attributes as well as numerical attributes formatted as text into numerals - wrote a regular expression that handles currency, CRISIL rankings, numbers formatted with commas, etc. and works well with decimals and signs.

3. AMFI dataset gives 12935 individual fund schemes. Restrict analysis to the top 10 fund families with the largest assets under management (the individual schemes in these families may still have very little assets under manamgent) as listed on MoneyControl.com AMC Asset Monitor. Therefore have only 1296 schems from 10 fund families in the final dataset.
4. Define and compute additional normalized metrics for each such as:

   a. Risk score based on MoneyControl risk rating - between 0 and 1

   b. CRISIL Rating (accreditation agency rating) depicting trustworthiness of fund scheme - between 0 and 1

   c. Ratio of AUM for fund scheme relative to AUM of fund family to which it belongs depicting the confidence that the fund family has in the scheme - between 0 and 1

   d. Fund performance relative to category performance - either 0 (if fund performance less than category performance) or 1 (if fund performance greater than category performance) (calculated for each time horizon)

   e. Volatility in fund scheme's category as ratio of category's worst to best performance - between 0 and 1 (calculated for each time horizon)

5. Use an Imputer for preprocessing and replace any NaN or missing values with 0.

Label Generation

Compute Expectation for each time horizon (1m, 3m, 6, 1y, 2y, 3y, 5y) based on the 5 metrics defined above to get an expected value between 0 and 1. Then round() the value to either 0 and 1 and use it as the label for binary classification where 1 represents a good investment option and 0 otherwise.

Feature Selection and Classification

Use the following as the features for our binary classification task:

1. Scheme Risk
2. CRISIL Rating
3. Fund Family AUM
4. Scheme AUM
5. Latest Net Asset Value
6. Minimum Investment for scheme
7. Latest Dividend
8. Scheme Bonus
9. Fund Return in any time frame
10. Category Average Return in any time frame

On further analysis, realize that only a few funds have paid out any dividends or issued bonuses, therefore these are unimportant features and can be eliminated. Transform features into a Pandas dataframe for analysis.

Separate ~23% data as test data and ~77% as training data. Perform 10-fold cross validation on Random Forest Classification with 1 to 40 trees in the forest, on training data for each individual time frame. Use box plots to visualize the results of cross-validation and pick an ideal estimator size. Train our Random Forest Classifier with the ideal estimator on the training data and check it's performance on test data by predicting labels and comparing them with the pre-assigned labels. Also generate feature importance charts from random forest classification to educate our users about the features to look at for any time horizon.

Visualization, Results and Interpretation

Obtain the following classification scores for each time frame:

1 month : 0.966216216216

3 month : 0.962837837838

6 month : 0.942567567568

1 year : 0.989864864865

2 year : 0.966216216216

3 year : 0.942567567568

5 year : 0.956081081081

The scores appear to be very high because of the class imbalance problem.

Out of a total of 1296 data points, only these many are labeled as good, the rest are bad:

1 month : 54 good / 1296 total

3 month : 52 good / 1296 total

6 month : 61 good / 1296 total

1 year : 72 good / 1296 total

2 year : 120 good / 1296 total

3 year : 139 good / 1296 total

5 year : 107 good / 1296 total

This means that if the classifier blindly assigned zeros to every data point, it would still produce a good score just because it correctly labelled bad data points as bad by chance. Observe this by calculating the precision, recall and plotting Receiver Operating Characteristics (ROC curves). The problem can be solved in two ways - either by reducing the number of bad samples (not recommended for this particular scenario) or by increasing the good samples (which can be done by duplicating the good samples). The model trained after making these changes would perform better on unseen samples. Solve the Class Imbalance problem and train a better model as a practice problem.

Lab Experiment to be done by students:

1. Load Dataset from AMFI and MoneyControl.com
2. Pre-process Data individual fund scheme wise to compute normalized metrics and process missing values.
3. Generate Labels at each time horizon based on 5 metrics such as Risk Score, CRISIL Rating, AUM Ratio, Fund to category performance, Volatility.
4. Select features for binary classification task.
5. Interpretation and Visualization Results by calculating the precision, recall and plotting Receiver Operating Characteristics (ROC curves).

FMC - LAB-4

Name: Sarvagya Singh

SAP: 60009200030

Div/Batch: K/K1

In [346]:

```
# pip install beautifulsoup4    # Download and install beautiful soup 4
# pip install lxml              # Download and install lxml for its XML and HTML parser
# pip install requests          # Download and install Python requests module
# pip install seaborn           # Download and install Seaborn for visualizations

from bs4 import BeautifulSoup
import numpy as np
import pandas as pd

import requests
import sys
import re

import sklearn
import sklearn.cross_validation
from sklearn.ensemble import RandomForestClassifier
from sklearn.cross_validation import cross_val_score

%matplotlib inline
import matplotlib
import matplotlib.pyplot as plt
import seaborn as sns

# from pyspark import SparkConf, SparkContext, SQLContext
# from pyspark.sql.types import *
#
# conf = SparkConf().setAppName('Project')
# sc = SparkContext(conf=conf)
# sqlContext = SQLContext(sc)

money_control_root = 'http://www.moneycontrol.com'
```

In [2]:

```
# Get 10 mutual fund families with the highest Assets under Management from Money Control
markup = requests.get(money_control_root + '/mutual-funds/amc-assets-monitor').text

# make the soup
soup = BeautifulSoup(markup, "lxml")

# the table that contains the required data
table = soup.find_all('table', attrs = {"class": "tblfund1"})[0]

# get the first ten rows in this table, excluding
# the first row as it has only header information
rows = table.find_all('tr')[1:11]


# fund_families_schema = StructType([
#     StructField("fund_family", StringType(), True),
#     StructField("fund_family_url", StringType(), True),
#     StructField("fund_family_aum", StringType(), True)
# ])

# Fund Family and Assets under Management (Rs. Cr.) for the top 10 mutual fund families
fund_families = []
for r in rows:
```

```
    ff_dict = {
        'fund_family_name': unicode( r.contents[1].a.string ),
        'fund_family_url' : unicode( money_control_root + r.contents[1].a.attrs['href']
),
        'fund_family_aum' : unicode( r.contents[5].string ),
        'fund_family_shortcode' : unicode( money_control_root + r.contents[1].a.attrs['h
ref'] ).split('/')[-1]
    }

    fund_families.append( ff_dict )
```

In [3]:

```
print( fund_families )
```

```
[{'fund_family_aum': u'178,373', 'fund_family_name': u'HDFC Mutual Fund', 'fund_family_ur
l': u'http://www.moneycontrol.com/mutual-funds/amc-details/HD', 'fund_family_shortcode':
u'HD'}, {'fund_family_aum': u'172,154', 'fund_family_name': u'ICICI Prudential Mutual Fun
d', 'fund_family_url': u'http://www.moneycontrol.com/mutual-funds/amc-details/PI', 'fund_
family_shortcode': u'PI'}, {'fund_family_aum': u'156,948', 'fund_family_name': u'Reliance
Mutual Fund', 'fund_family_url': u'http://www.moneycontrol.com/mutual-funds/amc-details/R
C', 'fund_family_shortcode': u'RC'}, {'fund_family_aum': u'136,561', 'fund_family_name':
u'Birla Sun Life Mutual Fund', 'fund_family_url': u'http://www.moneycontrol.com/mutual-fu
nds/amc-details/BS', 'fund_family_shortcode': u'BS'}, {'fund_family_aum': u'106,129', 'fu
nd_family_name': u'UTI Mutual Fund', 'fund_family_url': u'http://www.moneycontrol.com/mut
ual-funds/amc-details/UT', 'fund_family_shortcode': u'UT'}, {'fund_family_aum': u'100,055
', 'fund_family_name': u'SBI Mutual Fund', 'fund_family_url': u'http://www.moneycontrol.c
om/mutual-funds/amc-details/SB', 'fund_family_shortcode': u'SB'}, {'fund_family_aum': u'7
0,780', 'fund_family_name': u'Franklin Templeton Mutual Fund', 'fund_family_url': u'http:
//www.moneycontrol.com/mutual-funds/amc-details/TE', 'fund_family_shortcode': u'TE'}, {'f
und_family_aum': u'54,902', 'fund_family_name': u'Kotak Mahindra Mutual Fund', 'fund_fami
ly_url': u'http://www.moneycontrol.com/mutual-funds/amc-details/KM', 'fund_family_shortco
de': u'KM'}, {'fund_family_aum': u'54,715', 'fund_family_name': u'IDFC Mutual Fund', 'fun
d_family_url': u'http://www.moneycontrol.com/mutual-funds/amc-details/AG', 'fund_family_s
hortcode': u'AG'}, {'fund_family_aum': u'38,099', 'fund_family_name': u'DSP BlackRock Mut
ual Fund', 'fund_family_url': u'http://www.moneycontrol.com/mutual-funds/amc-details/DS',
'fund_family_shortcode': u'DS'}]
```

In [284]:

```
# For each fund family, get a list of all fund schemes along with other details
fund_schemes = []
for fund in fund_families:

    markup = requests.get( fund['fund_family_url'] ).text

    soup = BeautifulSoup(markup, "lxml")

    rows = soup.select('.FL.MT10.boxBg table tr')[1:-1]

    for r in rows:
        data_elems      = r.find_all('td')

        category_name   = ''
        scheme_aum      = ''
        category_url    = ''

        try:
            category_name   = unicode( data_elems[2].a.string )
            category_url    = money_control_root + data_elems[2].a.attrs['href']

        except AttributeError:
            category_name   = u'None'
            category_url    = u'None'

        try:
            scheme_aum = unicode( data_elems[5].string )
        except AttributeError:
            scheme_aum = u'None'

        fscheme_dict    = {
```

```
            'fund_family_name'      : fund['fund_family_name'],
            'fund_family_url'       : fund['fund_family_url' ],
            'fund_family_aum'       : fund['fund_family_aum' ],
            'fund_family_shortcode' : fund['fund_family_shortcode'],
            'scheme_name'           : unicode( data_elems[0].a.string ),
            'scheme_url'            : money_control_root + data_elems[0].a.attrs['href']
,
            'crisil_rating'         : unicode( data_elems[1].a.string ),
            'category'              : category_name,
            'category_url'          : category_url,
            'latest_nav'            : unicode( data_elems[3].string ),
            '1yr_return'            : u'None' if unicode( data_elems[4].string ) == u'-
-' else unicode( data_elems[4].string ),
            'scheme_aum'            : scheme_aum
        }

        fund_schemes.append( fscheme_dict )
```

In [285]:

```
print( len( fund_schemes ), '\n\n', fund_schemes[:10])
```

```
(1296, '\n\n', [{'fund_family_aum': u'178,373', 'fund_family_shortcode': u'HD', 'scheme_u
rl': 'http://www.moneycontrol.com/mutual-funds/nav/hdfc-arbitrage-fund-direct-plan/MHD117
1', 'latest_nav': u'18.40', 'fund_family_name': u'HDFC Mutual Fund', 'fund_family_url': u
'http://www.moneycontrol.com/mutual-funds/amc-details/HD', 'category': u'Arbitrage & Arbi
trage Plus', 'scheme_name': u'HDFC Arbitrage Fund - Direct (G)', 'scheme_aum': u'3.67', '
1yr_return': u'7.3', 'category_url': 'http://www.moneycontrol.com/mutual-funds/performanc
e-tracker/returns/arbitrage-and-arbitrage-plus.html', 'crisil_rating': u'Not Ranked'}, {'
fund_family_aum': u'178,373', 'fund_family_shortcode': u'HD', 'scheme_url': 'http://www.m
oneycontrol.com/mutual-funds/nav/hdfc-arbitrage-fund-retail-plan/MHD225', 'latest_nav': u
'18.19', 'fund_family_name': u'HDFC Mutual Fund', 'fund_family_url': u'http://www.moneyco
ntrol.com/mutual-funds/amc-details/HD', 'category': u'Arbitrage & Arbitrage Plus', 'schem
e_name': u'HDFC Arbitrage Fund - RP (G)', 'scheme_aum': u'17.00', '1yr_return': u'6.7', '
category_url': 'http://www.moneycontrol.com/mutual-funds/performance-tracker/returns/arbi
trage-and-arbitrage-plus.html', 'crisil_rating': u'Not Ranked'}, {'fund_family_aum': u'17
8,373', 'fund_family_shortcode': u'HD', 'scheme_url': 'http://www.moneycontrol.com/mutual
-funds/nav/hdfc-arbitrage-fund-wholesale-plan/MHD228', 'latest_nav': u'18.51', 'fund_fami
ly_name': u'HDFC Mutual Fund', 'fund_family_url': u'http://www.moneycontrol.com/mutual-fu
nds/amc-details/HD', 'category': u'Arbitrage & Arbitrage Plus', 'scheme_name': u'HDFC Arb
itrage Fund - WP (G)', 'scheme_aum': u'1,492.91', '1yr_return': u'6.8', 'category_url': '
http://www.moneycontrol.com/mutual-funds/performance-tracker/returns/arbitrage-and-arbitr
age-plus.html', 'crisil_rating': u'Not Ranked'}, {'fund_family_aum': u'178,373', 'fund_fa
mily_shortcode': u'HD', 'scheme_url': 'http://www.moneycontrol.com/mutual-funds/nav/hdfc-
arbitrage-fund-wholesale-plan-direct-plan/MHD2147', 'latest_nav': u'11.61', 'fund_family_
name': u'HDFC Mutual Fund', 'fund_family_url': u'http://www.moneycontrol.com/mutual-funds
/amc-details/HD', 'category': u'Arbitrage & Arbitrage Plus', 'scheme_name': u'HDFC Arbitr
age Fund - WP - DP (G)', 'scheme_aum': u'183.64', '1yr_return': u'7.3', 'category_url': '
http://www.moneycontrol.com/mutual-funds/performance-tracker/returns/arbitrage-and-arbitr
age-plus.html', 'crisil_rating': u'Not Ranked'}, {'fund_family_aum': u'178,373', 'fund_fa
mily_shortcode': u'HD', 'scheme_url': 'http://www.moneycontrol.com/mutual-funds/nav/hdfc-
balanced-fund/MHD002', 'latest_nav': u'106.40', 'fund_family_name': u'HDFC Mutual Fund',
'fund_family_url': u'http://www.moneycontrol.com/mutual-funds/amc-details/HD', 'category'
: u'Balanced', 'scheme_name': u'HDFC Balanced Fund (G)', 'scheme_aum': u'4,301.87', '1yr_
return': u'-2.6', 'category_url': 'http://www.moneycontrol.com/mutual-funds/performance-t
racker/returns/balanced.html', 'crisil_rating': u'Rank 2'}, {'fund_family_aum': u'178,373
', 'fund_family_shortcode': u'HD', 'scheme_url': 'http://www.moneycontrol.com/mutual-fund
s/nav/hdfc-balanced-fund-direct-plan/MHD1173', 'latest_nav': u'109.01', 'fund_family_name
': u'HDFC Mutual Fund', 'fund_family_url': u'http://www.moneycontrol.com/mutual-funds/amc
-details/HD', 'category': u'Balanced', 'scheme_name': u'HDFC Balanced Fund - Direct (G)',
'scheme_aum': u'430.24', '1yr_return': u'-1.6', 'category_url': 'http://www.moneycontrol.
com/mutual-funds/performance-tracker/returns/balanced.html', 'crisil_rating': u'Not Ranke
d'}, {'fund_family_aum': u'178,373', 'fund_family_shortcode': u'HD', 'scheme_url': 'http:
//www.moneycontrol.com/mutual-funds/nav/hdfc-banking-and-psu-debt-fund-regular-plan/MHD20
59', 'latest_nav': u'12.01', 'fund_family_name': u'HDFC Mutual Fund', 'fund_family_url':
u'http://www.moneycontrol.com/mutual-funds/amc-details/HD', 'category': u'Ultra Short Ter
m Debt', 'scheme_name': u'HDFC Banking & PSU Debt - Reg (G)', 'scheme_aum': u'40.10', '1y
r_return': u'8.8', 'category_url': 'http://www.moneycontrol.com/mutual-funds/performance-
tracker/returns/ultra-short-term-debt.html', 'crisil_rating': u'Not Ranked'}, {'fund_fami
ly_aum': u'178,373', 'fund_family_shortcode': u'HD', 'scheme_url': 'http://www.moneycontr
ol.com/mutual-funds/nav/hdfc-banking-and-psu-debt-fund-direct-plan/MHD2061', 'latest_nav'
: u'12.03', 'fund_family_name': u'HDFC Mutual Fund', 'fund_family_url': u'http://www.mone
```

ycontrol.com/mutual-funds/amc-details/HD', 'category': u'Ultra Short Term Debt', 'scheme_name': u'HDFC Banking & PSU Debt -Direct (G)', 'scheme_aum': u'115.68', '1yr_return': u'8.9', 'category_url': 'http://www.moneycontrol.com/mutual-funds/performance-tracker/returns/ultra-short-term-debt.html', 'crisil_rating': u'Not Ranked'}, {'fund_family_aum': u'178,373', 'fund_family_shortcode': u'HD', 'scheme_url': 'http://www.moneycontrol.com/mutual-funds/nav/hdfc-capital-builder-fund-direct-plan/MHD1148', 'latest_nav': u'197.53', 'fund_family_name': u'HDFC Mutual Fund', 'fund_family_url': u'http://www.moneycontrol.com/mutual-funds/amc-details/HD', 'category': u'Diversified Equity', 'scheme_name': u'HDFC Capital Builder - Direct (G)', 'scheme_aum': u'91.02', '1yr_return': u'-3.7', 'category_url': 'http://www.moneycontrol.com/mutual-funds/performance-tracker/returns/diversified-equity.html', 'crisil_rating': u'Not Ranked'}, {'fund_family_aum': u'178,373', 'fund_family_shortcode': u'HD', 'scheme_url': 'http://www.moneycontrol.com/mutual-funds/nav/hdfc-capital-builder-fund/MZU016', 'latest_nav': u'193.86', 'fund_family_name': u'HDFC Mutual Fund', 'fund_family_url': u'http://www.moneycontrol.com/mutual-funds/amc-details/HD', 'category': u'Diversified Equity', 'scheme_name': u'HDFC Capital Builder Fund (G)', 'scheme_aum': u'941.53', '1yr_return': u'-4.5', 'category_url': 'http://www.moneycontrol.com/mutual-funds/performance-tracker/returns/diversified-equity.html', 'crisil_rating': u'Rank 2'}])

In [286]:

```
for idx, scheme in enumerate(fund_schemes):
    # Read the page at the URL for each scheme
    markup = requests.get( scheme['scheme_url'] ).text
    soup = BeautifulSoup(markup, "lxml")


    # Riskometer (Risk Rating)
    scheme['scheme_risk_text'] = unicode(soup.select('.header .MT10 .toplft_cl3 p.avgbgtit')[0].string )

    # Scheme Plan and Scheme Option
    scheme_plan_option_data    = [unicode( x.string ).strip() for x in soup.select('#plannname_frm .FL span')]

    [scheme['scheme_plan'],
     scheme['scheme_option'] ] = scheme_plan_option_data if scheme_plan_option_data else [u'None', u'None']


    # From the Investment Info section, collect scheme fund type,
    # benchmark name, minimum investment required for this scheme,
    # last dividend or bonus, if paid else NA
    sub_soup = soup.select('.mainCont .tog_cont .MT20 .FL td')
    [scheme['scheme_fund_type'],
     scheme['scheme_benchmark'],
     scheme['scheme_min_investment'],
     scheme['scheme_last_dividend'],
     scheme['scheme_bonus'] ] = [
        unicode(x.string).strip() if( x.string and unicode(x.string).strip() != u'N.A.') else u'None' for x in [
            sub_soup[0],
            sub_soup[3],
            sub_soup[5],
            sub_soup[6],
            sub_soup[7]
        ]
    ]



    # From the performance section, gather
    # Fund Returns, Category Avg, Difference of Fund Returns and Category Returns
    # Best of category and worst of category
    sub_soup = soup.select('.mainCont .tog_cont table')[0]

    # Get the relevant table rows containing this information
    rows = [row for row in sub_soup if not row.string and unicode(row).strip()][1:]

    for row in rows:
        row_attrs = [x for x in row.children if unicode(x).strip()]
```

```
        row_name  = unicode(row_attrs[0].string).strip().lower()

        # fund returns
        if row_name == 'fund returns':
            scheme['fund_ret_1m']      = u'None' if unicode( row_attrs[1].string ) == u'
--' else unicode( row_attrs[1].string )
            scheme['fund_ret_3m']      = u'None' if unicode( row_attrs[2].string ) == u'
--' else unicode( row_attrs[2].string )
            scheme['fund_ret_6m']      = u'None' if unicode( row_attrs[3].string ) == u'
--' else unicode( row_attrs[3].string )
            scheme['fund_ret_1y']      = u'None' if unicode( row_attrs[4].string ) == u'
--' else unicode( row_attrs[4].string )
            scheme['fund_ret_2y']      = u'None' if unicode( row_attrs[5].string ) == u'
--' else unicode( row_attrs[5].string )
            scheme['fund_ret_3y']      = u'None' if unicode( row_attrs[6].string ) == u'
--' else unicode( row_attrs[6].string )
            scheme['fund_ret_5y']      = u'None' if unicode( row_attrs[7].string ) == u'
--' else unicode( row_attrs[7].string )

        # category avg
        if row_name == 'category avg':
            scheme['cat_avg_ret_1m']   = u'None' if unicode( row_attrs[1].string ) == u'
--' else unicode( row_attrs[1].string )
            scheme['cat_avg_ret_3m']   = u'None' if unicode( row_attrs[2].string ) == u'
--' else unicode( row_attrs[2].string )
            scheme['cat_avg_ret_6m']   = u'None' if unicode( row_attrs[3].string ) == u'
--' else unicode( row_attrs[3].string )
            scheme['cat_avg_ret_1y']   = u'None' if unicode( row_attrs[4].string ) == u'
--' else unicode( row_attrs[4].string )
            scheme['cat_avg_ret_2y']   = u'None' if unicode( row_attrs[5].string ) == u'
--' else unicode( row_attrs[5].string )
            scheme['cat_avg_ret_3y']   = u'None' if unicode( row_attrs[6].string ) == u'
--' else unicode( row_attrs[6].string )
            scheme['cat_avg_ret_5y']   = u'None' if unicode( row_attrs[7].string ) == u'
--' else unicode( row_attrs[7].string )

        # difference of fund returns and category returns
        if row_name == 'difference of fund returns and category returns':
            scheme['diff_fund_cat_1m'] = u'None' if unicode( row_attrs[1].string ) == u'
--' else unicode( row_attrs[1].string )
            scheme['diff_fund_cat_3m'] = u'None' if unicode( row_attrs[2].string ) == u'
--' else unicode( row_attrs[2].string )
            scheme['diff_fund_cat_6m'] = u'None' if unicode( row_attrs[3].string ) == u'
--' else unicode( row_attrs[3].string )
            scheme['diff_fund_cat_1y'] = u'None' if unicode( row_attrs[4].string ) == u'
--' else unicode( row_attrs[4].string )
            scheme['diff_fund_cat_2y'] = u'None' if unicode( row_attrs[5].string ) == u'
--' else unicode( row_attrs[5].string )
            scheme['diff_fund_cat_3y'] = u'None' if unicode( row_attrs[6].string ) == u'
--' else unicode( row_attrs[6].string )
            scheme['diff_fund_cat_5y'] = u'None' if unicode( row_attrs[7].string ) == u'
--' else unicode( row_attrs[7].string )

        # best of category
        if row_name == 'best of category':
            scheme['cat_best_1m']      = u'None' if unicode( row_attrs[1].string ) == u'
--' else unicode( row_attrs[1].string )
            scheme['cat_best_3m']      = u'None' if unicode( row_attrs[2].string ) == u'
--' else unicode( row_attrs[2].string )
            scheme['cat_best_6m']      = u'None' if unicode( row_attrs[3].string ) == u'
--' else unicode( row_attrs[3].string )
            scheme['cat_best_1y']      = u'None' if unicode( row_attrs[4].string ) == u'
--' else unicode( row_attrs[4].string )
            scheme['cat_best_2y']      = u'None' if unicode( row_attrs[5].string ) == u'
--' else unicode( row_attrs[5].string )
            scheme['cat_best_3y']      = u'None' if unicode( row_attrs[6].string ) == u'
--' else unicode( row_attrs[6].string )
            scheme['cat_best_5y']      = u'None' if unicode( row_attrs[7].string ) == u'
--' else unicode( row_attrs[7].string )

        # worst of category
        if row_name == 'worst of category':
```

```
            scheme['cat_worst_1m']     = u'None' if unicode( row_attrs[1].string ) == u'
--' else unicode( row_attrs[1].string )
            scheme['cat_worst_3m']     = u'None' if unicode( row_attrs[2].string ) == u'
--' else unicode( row_attrs[2].string )
            scheme['cat_worst_6m']     = u'None' if unicode( row_attrs[3].string ) == u'
--' else unicode( row_attrs[3].string )
            scheme['cat_worst_1y']     = u'None' if unicode( row_attrs[4].string ) == u'
--' else unicode( row_attrs[4].string )
            scheme['cat_worst_2y']     = u'None' if unicode( row_attrs[5].string ) == u'
--' else unicode( row_attrs[5].string )
            scheme['cat_worst_3y']     = u'None' if unicode( row_attrs[6].string ) == u'
--' else unicode( row_attrs[6].string )
            scheme['cat_worst_5y']     = u'None' if unicode( row_attrs[7].string ) == u'
--' else unicode( row_attrs[7].string )

    # Print every 100th scheme to verify things are running smoothly
    if idx % 100 == 0:
        print( 'Scheme # {0}\n{1}\n\n\n'.format(idx, scheme) )
    else:
        print idx, ' ',
```

```
Scheme # 0
{'fund_family_aum': u'178,373', 'fund_family_shortcode': u'HD', 'scheme_plan': u'Direct',
'scheme_risk_text': u'MODERATELY LOW', 'fund_family_url': u'http://www.moneycontrol.com/m
utual-funds/amc-details/HD', 'cat_best_6m': u'4.0', 'diff_fund_cat_1y': u'1.2', 'cat_avg_
ret_3y': u'6.1', 'cat_avg_ret_3m': u'1.4', 'diff_fund_cat_1m': u'-0.1', 'diff_fund_cat_5y
': u'None', 'cat_worst_1m': u'0.4', 'cat_worst_1y': u'5.5', 'latest_nav': u'18.40', 'sche
me_last_dividend': u'None', 'scheme_min_investment': u'Rs.5000', 'diff_fund_cat_6m': u'0.
5', 'fund_ret_3y': u'8.0', 'fund_ret_2y': u'7.7', 'cat_worst_5y': u'7.7', 'cat_best_5y':
u'8.8', '1yr_return': u'7.3', 'category_url': 'http://www.moneycontrol.com/mutual-funds/p
erformance-tracker/returns/arbitrage-and-arbitrage-plus.html', 'cat_best_1m': u'1.7', 'di
ff_fund_cat_2y': u'2.0', 'cat_avg_ret_2y': u'5.7', 'fund_ret_6m': u'3.2', 'cat_best_1y':
u'8.2', 'cat_worst_6m': u'2.4', 'scheme_bonus': u'None', 'scheme_benchmark': u'CRISIL Liq
uid Fund', 'scheme_fund_type': u'Open-Ended', 'cat_worst_2y': u'5.5', 'scheme_url': 'http
://www.moneycontrol.com/mutual-funds/nav/hdfc-arbitrage-fund-direct-plan/MHD1171', 'fund_
family_name': u'HDFC Mutual Fund', 'scheme_aum': u'3.67', 'cat_avg_ret_5y': u'6.2', 'sche
me_name': u'HDFC Arbitrage Fund - Direct (G)', 'scheme_option': u'Growth', 'diff_fund_cat
_3y': u'1.9', 'cat_avg_ret_1m': u'0.6', 'diff_fund_cat_3m': u'0.2', 'cat_avg_ret_1y': u'6
.1', 'category': u'Arbitrage & Arbitrage Plus', 'cat_worst_3m': u'1.3', 'fund_ret_3m': u'
1.6', 'cat_avg_ret_6m': u'2.7', 'cat_worst_3y': u'7.5', 'fund_ret_5y': u'None', 'cat_best
_3y': u'9.3', 'fund_ret_1m': u'0.5', 'cat_best_2y': u'9.8', 'cat_best_3m': u'2.8', 'crisi
l_rating': u'Not Ranked', 'fund_ret_1y': u'7.3'}



1   2   3   4   5   6   7   8   9   10   11   12   13   14   15   16   17   18   19   20
21   22   23   24   25   26   27   28   29   30   31   32   33   34   35   36   37   38
39   40   41   42   43   44   45   46   47   48   49   50   51   52   53   54   55   56
57   58   59   60   61   62   63   64   65   66   67   68   69   70   71   72   73   74
75   76   77   78   79   80   81   82   83   84   85   86   87   88   89   90   91   92
93   94   95   96   97   98   99   Scheme # 100
{'fund_family_aum': u'178,373', 'fund_family_shortcode': u'HD', 'scheme_plan': u'Wholesal
e Plan', 'scheme_risk_text': u'LOW', 'fund_family_url': u'http://www.moneycontrol.com/mut
ual-funds/amc-details/HD', 'cat_best_6m': u'4.1', 'diff_fund_cat_1y': u'-26.7', 'cat_avg_
ret_3y': u'3.1', 'cat_avg_ret_3m': u'0.7', 'diff_fund_cat_1m': u'None', 'diff_fund_cat_5y
': u'None', 'cat_worst_1m': u'0.3', 'cat_worst_1y': u'-24.7', 'latest_nav': u'10.00', 'sc
heme_last_dividend': u'None', 'scheme_min_investment': u'Rs.10000000', 'diff_fund_cat_6m'
: u'None', 'fund_ret_3y': u'-3.3', 'fund_ret_2y': u'-9.2', 'cat_worst_5y': u'2.0', 'cat_b
est_5y': u'9.3', '1yr_return': u'-24.7', 'category_url': u'None', 'cat_best_1m': u'0.6',
'diff_fund_cat_2y': u'-12.2', 'cat_avg_ret_2y': u'3.0', 'fund_ret_6m': u'None', 'cat_best
_1y': u'8.6', 'cat_worst_6m': u'0.4', 'scheme_bonus': u'None', 'scheme_benchmark': u'CRIS
IL Liquid Fund', 'scheme_fund_type': u'Open-Ended', 'cat_worst_2y': u'-9.2', 'scheme_url'
: 'http://www.moneycontrol.com/mutual-funds/nav/hdfc-quarterly-interval-fund-plan-c-whole
sale-plan/MHD080', 'fund_family_name': u'HDFC Mutual Fund', 'scheme_aum': u'1.00', 'cat_a
vg_ret_5y': u'3.6', 'scheme_name': u'HDFC QIF - Plan C - WP (G)', 'scheme_option': u'Grow
th', 'diff_fund_cat_3y': u'-6.4', 'cat_avg_ret_1m': u'0.2', 'diff_fund_cat_3m': u'None',
'cat_avg_ret_1y': u'2.0', 'category': u'None', 'cat_worst_3m': u'0.1', 'fund_ret_3m': u'N
one', 'cat_avg_ret_6m': u'1.5', 'cat_worst_3y': u'-3.3', 'fund_ret_5y': u'None', 'cat_bes
t_3y': u'9.2', 'fund_ret_1m': u'None', 'cat_best_2y': u'9.1', 'cat_best_3m': u'1.9', 'cri
sil_rating': u'Not Ranked', 'fund_ret_1y': u'-24.7'}
```

101 102 103 104 105 106 107 108 109 110 111 112 113 114 115
116 117 118 119 120 121 122 123 124 125 126 127 128 129 130
131 132 133 134 135 136 137 138 139 140 141 142 143 144 145
146 147 148 149 150 151 152 153 154 155 156 157 158 159 160
161 162 163 164 165 166 167 168 169 170 171 172 173 174 175
176 177 178 179 180 181 182 183 184 185 186 187 188 189 190
191 192 193 194 195 196 197 198 199 Scheme # 200
{'fund_family_aum': u'172,154', 'fund_family_shortcode': u'PI', 'scheme_plan': u'Institut
ional Option I', 'scheme_risk_text': u'MODERATELY LOW', 'fund_family_url': u'http://www.m
oneycontrol.com/mutual-funds/amc-details/PI', 'cat_best_6m': u'0.7', 'diff_fund_cat_1y':
u'None', 'cat_avg_ret_3y': u'8.6', 'cat_avg_ret_3m': u'2.5', 'diff_fund_cat_1m': u'None',
'diff_fund_cat_5y': u'None', 'cat_worst_1m': u'0.00', 'cat_worst_1y': u'0.00', 'latest_na
v': u'10.01', 'scheme_last_dividend': u'None', 'scheme_min_investment': u'Rs.500', 'diff_
fund_cat_6m': u'None', 'fund_ret_3y': u'None', 'fund_ret_2y': u'None', 'cat_worst_5y': u'
0.00', 'cat_best_5y': u'8.0', '1yr_return': u'None', 'category_url': 'http://www.moneycon
trol.com/mutual-funds/performance-tracker/returns/.html', 'cat_best_1m': u'1.3', 'diff_fu
nd_cat_2y': u'None', 'cat_avg_ret_2y': u'7.3', 'fund_ret_6m': u'None', 'cat_best_1y': u'1
.5', 'cat_worst_6m': u'0.00', 'scheme_bonus': u'None', 'scheme_benchmark': u'CRISIL Liqui
d Fund', 'scheme_fund_type': u'Open-Ended', 'cat_worst_2y': u'0.00', 'scheme_url': 'http:
//www.moneycontrol.com/mutual-funds/nav/icici-prudential-flexible-income-plan-institution
al-option-i/MPI642', 'fund_family_name': u'ICICI Prudential Mutual Fund', 'scheme_aum': u
'4.05', 'cat_avg_ret_5y': u'8.0', 'scheme_name': u'ICICI Pru Flexi Inc -Inst - I', 'schem
e_option': u'Growth', 'diff_fund_cat_3y': u'None', 'cat_avg_ret_1m': u'4.0', 'diff_fund_c
at_3m': u'None', 'cat_avg_ret_1y': u'9.2', 'category': u'None', 'cat_worst_3m': u'0.00',
'fund_ret_3m': u'None', 'cat_avg_ret_6m': u'4.4', 'cat_worst_3y': u'0.00', 'fund_ret_5y':
u'None', 'cat_best_3y': u'8.0', 'fund_ret_1m': u'None', 'cat_best_2y': u'6.7', 'cat_best_
3m': u'0.1', 'crisil_rating': u'Not Ranked', 'fund_ret_1y': u'None'}

201 202 203 204 205 206 207 208 209 210 211 212 213 214 215
216 217 218 219 220 221 222 223 224 225 226 227 228 229 230
231 232 233 234 235 236 237 238 239 240 241 242 243 244 245
246 247 248 249 250 251 252 253 254 255 256 257 258 259 260
261 262 263 264 265 266 267 268 269 270 271 272 273 274 275
276 277 278 279 280 281 282 283 284 285 286 287 288 289 290
291 292 293 294 295 296 297 298 299 Scheme # 300
{'fund_family_aum': u'172,154', 'fund_family_shortcode': u'PI', 'scheme_plan': u'Regular'
, 'scheme_risk_text': u'MODERATE', 'fund_family_url': u'http://www.moneycontrol.com/mutua
l-funds/amc-details/PI', 'cat_best_6m': u'10.4', 'diff_fund_cat_1y': u'13.7', 'cat_avg_re
t_3y': u'7.9', 'cat_avg_ret_3m': u'0.4', 'diff_fund_cat_1m': u'-1.3', 'diff_fund_cat_5y':
u'None', 'cat_worst_1m': u'-0.9', 'cat_worst_1y': u'-5.8', 'latest_nav': u'15.21', 'schem
e_last_dividend': u'None', 'scheme_min_investment': u'Rs.5000', 'diff_fund_cat_6m': u'6.6
', 'fund_ret_3y': u'15.1', 'fund_ret_2y': u'14.8', 'cat_worst_5y': u'5.5', 'cat_best_5y':
u'10.8', '1yr_return': u'18.4', 'category_url': 'http://www.moneycontrol.com/mutual-funds
/performance-tracker/returns/mip-aggressive.html', 'cat_best_1m': u'4.3', 'diff_fund_cat_
2y': u'4.2', 'cat_avg_ret_2y': u'10.6', 'fund_ret_6m': u'8.1', 'cat_best_1y': u'23.4', 'c
at_worst_6m': u'-6.4', 'scheme_bonus': u'None', 'scheme_benchmark': u'CRISIL MIP Blended
Fund', 'scheme_fund_type': u'Close-Ended', 'cat_worst_2y': u'5.9', 'scheme_url': 'http://
www.moneycontrol.com/mutual-funds/nav/icici-prudential-multiple-yield-fund-plan-e/MPI946'
, 'fund_family_name': u'ICICI Prudential Mutual Fund', 'scheme_aum': u'55.00', 'cat_avg_r
et_5y': u'4.7', 'scheme_name': u'ICICI Pru Multiple Yield-Plan E (G)', 'scheme_option': u
'Growth', 'diff_fund_cat_3y': u'7.2', 'cat_avg_ret_1m': u'1.8', 'diff_fund_cat_3m': u'2.2
', 'cat_avg_ret_1y': u'4.7', 'category': u'MIP Aggressive', 'cat_worst_3m': u'-3.2', 'fun
d_ret_3m': u'2.6', 'cat_avg_ret_6m': u'1.5', 'cat_worst_3y': u'5.2', 'fund_ret_5y': u'Non
e', 'cat_best_3y': u'16.3', 'fund_ret_1m': u'0.5', 'cat_best_2y': u'19.4', 'cat_best_3m':
u'3.4', 'crisil_rating': u'Not Ranked', 'fund_ret_1y': u'18.4'}

301 302 303 304 305 306 307 308 309 310 311 312 313 314 315
316 317 318 319 320 321 322 323 324 325 326 327 328 329 330
331 332 333 334 335 336 337 338 339 340 341 342 343 344 345
346 347 348 349 350 351 352 353 354 355 356 357 358 359 360
361 362 363 364 365 366 367 368 369 370 371 372 373 374 375
376 377 378 379 380 381 382 383 384 385 386 387 388 389 390
391 392 393 394 395 396 397 398 399 Scheme # 400
{'fund_family_aum': u'156,948', 'fund_family_shortcode': u'RC', 'scheme_plan': u'Direct',
'scheme_risk_text': u'HIGH', 'fund_family_url': u'http://www.moneycontrol.com/mutual-fund
s/amc-details/RC', 'cat_best_6m': u'4.6', 'diff_fund_cat_1y': u'None', 'cat_avg_ret_3y':
u'10.6', 'cat_avg_ret_3m': u'-4.1', 'diff_fund_cat_1m': u'-2.7', 'diff_fund_cat_5y': u'No

ne', 'cat_worst_1m': u'-4.6', 'cat_worst_1y': u'-22.8', 'latest_nav': u'9.14', 'scheme_last_dividend': u'None', 'scheme_min_investment': u'Rs.5000', 'diff_fund_cat_6m': u'-4.1', 'fund_ret_3y': u'None', 'fund_ret_2y': u'None', 'cat_worst_5y': u'-6.1', 'cat_best_5y': u'19.6', '1yr_return': u'None', 'category_url': 'http://www.moneycontrol.com/mutual-funds/performance-tracker/returns/diversified-equity.html', 'cat_best_1m': u'9.7', 'diff_fund_cat_2y': u'None', 'cat_avg_ret_2y': u'10.8', 'fund_ret_6m': u'-9.2', 'cat_best_1y': u'3.1', 'cat_worst_6m': u'-18.5', 'scheme_bonus': u'None', 'scheme_benchmark': u'S&P BSE 200', 'scheme_fund_type': u'Close-Ended', 'cat_worst_2y': u'-5.8', 'scheme_url': 'http://www.moneycontrol.com/mutual-funds/nav/reliance-capital-builder-fund-iii-series-a-direct-plan/MRC1889', 'fund_family_name': u'Reliance Mutual Fund', 'scheme_aum': u'4.30', 'cat_avg_ret_5y': u'5.7', 'scheme_name': u'Reliance Capital Builder-III-Sr-A DP(G)', 'scheme_option': u'Growth', 'diff_fund_cat_3y': u'None', 'cat_avg_ret_1m': u'5.2', 'diff_fund_cat_3m': u'1.8', 'cat_avg_ret_1y': u'-7.7', 'category': u'Diversified Equity', 'cat_worst_3m': u'-12.4', 'fund_ret_3m': u'-2.3', 'cat_avg_ret_6m': u'-5.1', 'cat_worst_3y': u'-1.2', 'fund_ret_5y': u'None', 'cat_best_3y': u'30.9', 'fund_ret_1m': u'2.5', 'cat_best_2y': u'31.9', 'cat_best_3m': u'1.2', 'crisil_rating': u'Not Ranked', 'fund_ret_1y': u'None'}

401 402 403 404 405 406 407 408 409 410 411 412 413 414 415
416 417 418 419 420 421 422 423 424 425 426 427 428 429 430
431 432 433 434 435 436 437 438 439 440 441 442 443 444 445
446 447 448 449 450 451 452 453 454 455 456 457 458 459 460
461 462 463 464 465 466 467 468 469 470 471 472 473 474 475
476 477 478 479 480 481 482 483 484 485 486 487 488 489 490
491 492 493 494 495 496 497 498 499 Scheme # 500

{'fund_family_aum': u'136,561', 'fund_family_shortcode': u'BS', 'scheme_plan': u'Direct', 'scheme_risk_text': u'HIGH', 'fund_family_url': u'http://www.moneycontrol.com/mutual-funds/amc-details/BS', 'cat_best_6m': u'6.3', 'diff_fund_cat_1y': u'-0.6', 'cat_avg_ret_3y': u'6.7', 'cat_avg_ret_3m': u'-2.5', 'diff_fund_cat_1m': u'-7.0', 'diff_fund_cat_5y': u'None', 'cat_worst_1m': u'-8.0', 'cat_worst_1y': u'-28.2', 'latest_nav': u'24.94', 'scheme_last_dividend': u'None', 'scheme_min_investment': u'Rs.5000', 'diff_fund_cat_6m': u'-6.4', 'fund_ret_3y': u'14.3', 'fund_ret_2y': u'23.1', 'cat_worst_5y': u'-11.8', 'cat_best_5y': u'12.1', '1yr_return': u'-6.7', 'category_url': 'http://www.moneycontrol.com/mutual-funds/performance-tracker/returns/fund-of-funds-equity-oriented.html', 'cat_best_1m': u'8.0', 'diff_fund_cat_2y': u'14.2', 'cat_avg_ret_2y': u'8.9', 'fund_ret_6m': u'-11.3', 'cat_best_1y': u'4.0', 'cat_worst_6m': u'-26.5', 'scheme_bonus': u'None', 'scheme_benchmark': u'NIFTY 50', 'scheme_fund_type': u'Open-Ended', 'cat_worst_2y': u'-19.8', 'scheme_url': 'http://www.moneycontrol.com/mutual-funds/nav/birla-sun-life-5-star-multi-manager-fof-scheme-direct-plan/MIN395', 'fund_family_name': u'Birla Sun Life Mutual Fund', 'scheme_aum': u'0.18', 'cat_avg_ret_5y': u'5.5', 'scheme_name': u'Birla SL 5 STAR MM FoF - Direct (G)', 'scheme_option': u'Growth', 'diff_fund_cat_3y': u'7.6', 'cat_avg_ret_1m': u'0.9', 'diff_fund_cat_3m': u'-3.5', 'cat_avg_ret_1y': u'-6.1', 'category': u'Fund of Funds - Equity oriented', 'cat_worst_3m': u'-12.5', 'fund_ret_3m': u'-6.0', 'cat_avg_ret_6m': u'-4.9', 'cat_worst_3y': u'-17.4', 'fund_ret_5y': u'None', 'cat_best_3y': u'19.3', 'fund_ret_1m': u'-6.1', 'cat_best_2y': u'23.1', 'cat_best_3m': u'5.3', 'crisil_rating': u'Not Ranked', 'fund_ret_1y': u'-6.7'}

501 502 503 504 505 506 507 508 509 510 511 512 513 514 515
516 517 518 519 520 521 522 523 524 525 526 527 528 529 530
531 532 533 534 535 536 537 538 539 540 541 542 543 544 545
546 547 548 549 550 551 552 553 554 555 556 557 558 559 560
561 562 563 564 565 566 567 568 569 570 571 572 573 574 575
576 577 578 579 580 581 582 583 584 585 586 587 588 589 590
591 592 593 594 595 596 597 598 599 Scheme # 600

{'fund_family_aum': u'136,561', 'fund_family_shortcode': u'BS', 'scheme_plan': u'Regular', 'scheme_risk_text': u'HIGH', 'fund_family_url': u'http://www.moneycontrol.com/mutual-funds/amc-details/BS', 'cat_best_6m': u'-2.4', 'diff_fund_cat_1y': u'None', 'cat_avg_ret_3y': u'3.1', 'cat_avg_ret_3m': u'-4.0', 'diff_fund_cat_1m': u'0.6', 'diff_fund_cat_5y': u'None', 'cat_worst_1m': u'1.9', 'cat_worst_1y': u'-20.9', 'latest_nav': u'9.71', 'scheme_last_dividend': u'None', 'scheme_min_investment': u'Rs.5000', 'diff_fund_cat_6m': u'None', 'fund_ret_3y': u'None', 'fund_ret_2y': u'None', 'cat_worst_5y': u'0.00', 'cat_best_5y': u'None', '1yr_return': u'None', 'category_url': 'http://www.moneycontrol.com/mutual-funds/performance-tracker/returns/rgess.html', 'cat_best_1m': u'8.8', 'diff_fund_cat_2y': u'None', 'cat_avg_ret_2y': u'4.5', 'fund_ret_6m': u'None', 'cat_best_1y': u'-7.7', 'cat_worst_6m': u'-13.3', 'scheme_bonus': u'None', 'scheme_benchmark': u'NIFTY 100', 'scheme_fund_type': u'Close-Ended', 'cat_worst_2y': u'7.0', 'scheme_url': 'http://www.moneycontrol.com/mutual-funds/nav/birla-sun-life-focused-equity-fund-series-6-regular-plan/MBS2120', 'fund_family_name': u'Birla Sun Life Mutual Fund', 'scheme_aum': u'58.00', 'cat_avg_ret_5y': u'None', 'scheme_name': u'Birla SL Focused Equity-Sr 6-RP (G)', 'scheme_option': u'Growth',

'diff_fund_cat_3y': u'None', 'cat_avg_ret_1m': u'5.8', 'diff_fund_cat_3m': u'3.0', 'cat_a
vg_ret_1y': u'-10.9', 'category': u'RGESS', 'cat_worst_3m': u'-8.6', 'fund_ret_3m': u'-1.
0', 'cat_avg_ret_6m': u'-7.0', 'cat_worst_3y': u'9.7', 'fund_ret_5y': u'None', 'cat_best_
3y': u'14.4', 'fund_ret_1m': u'6.4', 'cat_best_2y': u'14.2', 'cat_best_3m': u'-0.6', 'cri
sil_rating': u'Not Ranked', 'fund_ret_1y': u'None'}

601 602 603 604 605 606 607 608 609 610 611 612 613 614 615
616 617 618 619 620 621 622 623 624 625 626 627 628 629 630
631 632 633 634 635 636 637 638 639 640 641 642 643 644 645
646 647 648 649 650 651 652 653 654 655 656 657 658 659 660
661 662 663 664 665 666 667 668 669 670 671 672 673 674 675
676 677 678 679 680 681 682 683 684 685 686 687 688 689 690
691 692 693 694 695 696 697 698 699 Scheme # 700

{'fund_family_aum': u'136,561', 'fund_family_shortcode': u'BS', 'scheme_plan': u'Regular'
, 'scheme_risk_text': u'MODERATE', 'fund_family_url': u'http://www.moneycontrol.com/mutua
l-funds/amc-details/BS', 'cat_best_6m': u'4.9', 'diff_fund_cat_1y': u'0.3', 'cat_avg_ret_
3y': u'7.9', 'cat_avg_ret_3m': u'1.0', 'diff_fund_cat_1m': u'0.6', 'diff_fund_cat_5y': u'
2.7', 'cat_worst_1m': u'-0.2', 'cat_worst_1y': u'-1.5', 'latest_nav': u'39.35', 'scheme_l
ast_dividend': u'None', 'scheme_min_investment': u'Rs.5000', 'diff_fund_cat_6m': u'-0.2',
'fund_ret_3y': u'8.8', 'fund_ret_2y': u'9.4', 'cat_worst_5y': u'6.2', 'cat_best_5y': u'10
.1', '1yr_return': u'4.0', 'category_url': 'http://www.moneycontrol.com/mutual-funds/perf
ormance-tracker/returns/mip-conservative.html', 'cat_best_1m': u'2.9', 'diff_fund_cat_2y'
: u'0.5', 'cat_avg_ret_2y': u'8.9', 'fund_ret_6m': u'1.5', 'cat_best_1y': u'10.1', 'cat_w
orst_6m': u'-0.4', 'scheme_bonus': u'None', 'scheme_benchmark': u'CRISIL MIP Blended Fund
', 'scheme_fund_type': u'Open-Ended', 'cat_worst_2y': u'6.7', 'scheme_url': 'http://www.m
oneycontrol.com/mutual-funds/nav/birla-sun-life-mip/MBS019', 'fund_family_name': u'Birla
Sun Life Mutual Fund', 'scheme_aum': u'106.06', 'cat_avg_ret_5y': u'5.3', 'scheme_name':
u'Birla Sun Life MIP (G)', 'scheme_option': u'Growth', 'diff_fund_cat_3y': u'0.9', 'cat_a
vg_ret_1m': u'1.8', 'diff_fund_cat_3m': u'-0.1', 'cat_avg_ret_1y': u'3.7', 'category': u'
MIP Conservative', 'cat_worst_3m': u'-0.1', 'fund_ret_3m': u'0.9', 'cat_avg_ret_6m': u'1.
7', 'cat_worst_3y': u'5.4', 'fund_ret_5y': u'8.0', 'cat_best_3y': u'12.4', 'fund_ret_1m':
u'2.4', 'cat_best_2y': u'13.6', 'cat_best_3m': u'2.6', 'crisil_rating': u'Not Ranked', 'f
und_ret_1y': u'4.0'}

701 702 703 704 705 706 707 708 709 710 711 712 713 714 715
716 717 718 719 720 721 722 723 724 725 726 727 728 729 730
731 732 733 734 735 736 737 738 739 740 741 742 743 744 745
746 747 748 749 750 751 752 753 754 755 756 757 758 759 760
761 762 763 764 765 766 767 768 769 770 771 772 773 774 775
776 777 778 779 780 781 782 783 784 785 786 787 788 789 790
791 792 793 794 795 796 797 798 799 Scheme # 800

{'fund_family_aum': u'106,129', 'fund_family_shortcode': u'UT', 'scheme_plan': u'Regular'
, 'scheme_risk_text': u'MODERATELY LOW', 'fund_family_url': u'http://www.moneycontrol.com
/mutual-funds/amc-details/UT', 'cat_best_6m': u'2.1', 'diff_fund_cat_1y': u'-3.1', 'cat_a
vg_ret_3y': u'13.5', 'cat_avg_ret_3m': u'-2.2', 'diff_fund_cat_1m': u'0.8', 'diff_fund_ca
t_5y': u'2.0', 'cat_worst_1m': u'-0.4', 'cat_worst_1y': u'-20.4', 'latest_nav': u'45.69',
'scheme_last_dividend': u'None', 'scheme_min_investment': u'Rs.5000', 'diff_fund_cat_6m':
u'-2.3', 'fund_ret_3y': u'15.3', 'fund_ret_2y': u'13.6', 'cat_worst_5y': u'2.3', 'cat_bes
t_5y': u'15.9', '1yr_return': u'-11.1', 'category_url': 'http://www.moneycontrol.com/mutu
al-funds/performance-tracker/returns/large-cap.html', 'cat_best_1m': u'8.9', 'diff_fund_c
at_2y': u'1.7', 'cat_avg_ret_2y': u'11.9', 'fund_ret_6m': u'-6.9', 'cat_best_1y': u'15.5'
, 'cat_worst_6m': u'-10.4', 'scheme_bonus': u'1:1 (Jun-04-2009)', 'scheme_benchmark': u'S
&P BSE 100', 'scheme_fund_type': u'Open-Ended', 'cat_worst_2y': u'1.2', 'scheme_url': 'ht
tp://www.moneycontrol.com/mutual-funds/nav/uti-top-100-fund/MUT010', 'fund_family_name':
u'UTI Mutual Fund', 'scheme_aum': u'862.97', 'cat_avg_ret_5y': u'7.0', 'scheme_name': u'U
TI Top 100 Fund (G)', 'scheme_option': u'Growth', 'diff_fund_cat_3y': u'1.8', 'cat_avg_re
t_1m': u'4.7', 'diff_fund_cat_3m': u'0.1', 'cat_avg_ret_1y': u'-8.0', 'category': u'Large
Cap', 'cat_worst_3m': u'-7.7', 'fund_ret_3m': u'-2.1', 'cat_avg_ret_6m': u'-4.6', 'cat_wo
rst_3y': u'6.0', 'fund_ret_5y': u'9.0', 'cat_best_3y': u'26.2', 'fund_ret_1m': u'5.5', 'c
at_best_2y': u'30.3', 'cat_best_3m': u'4.9', 'crisil_rating': u'Rank 3', 'fund_ret_1y': u
'-11.1'}

801 802 803 804 805 806 807 808 809 810 811 812 813 814 815
816 817 818 819 820 821 822 823 824 825 826 827 828 829 830
831 832 833 834 835 836 837 838 839 840 841 842 843 844 845
846 847 848 849 850 851 852 853 854 855 856 857 858 859 860

861 862 863 864 865 866 867 868 869 870 871 872 873 874 875
876 877 878 879 880 881 882 883 884 885 886 887 888 889 890
891 892 893 894 895 896 897 898 899 Scheme # 900
{'fund_family_aum': u'100,055', 'fund_family_shortcode': u'SB', 'scheme_plan': u'Regular'
, 'scheme_risk_text': u'MODERATELY HIGH', 'fund_family_url': u'http://www.moneycontrol.co
m/mutual-funds/amc-details/SB', 'cat_best_6m': u'4.6', 'diff_fund_cat_1y': u'7.6', 'cat_a
vg_ret_3y': u'10.6', 'cat_avg_ret_3m': u'-4.1', 'diff_fund_cat_1m': u'4.4', 'diff_fund_ca
t_5y': u'6.9', 'cat_worst_1m': u'-4.6', 'cat_worst_1y': u'-22.8', 'latest_nav': u'32.59',
'scheme_last_dividend': u'None', 'scheme_min_investment': u'Rs.1000', 'diff_fund_cat_6m':
u'5.7', 'fund_ret_3y': u'22.5', 'fund_ret_2y': u'25.2', 'cat_worst_5y': u'-6.1', 'cat_bes
t_5y': u'19.6', '1yr_return': u'-0.1', 'category_url': 'http://www.moneycontrol.com/mutua
l-funds/performance-tracker/returns/diversified-equity.html', 'cat_best_1m': u'9.7', 'dif
f_fund_cat_2y': u'14.4', 'cat_avg_ret_2y': u'10.8', 'fund_ret_6m': u'0.6', 'cat_best_1y':
u'3.1', 'cat_worst_6m': u'-18.5', 'scheme_bonus': u'None', 'scheme_benchmark': u'S&P BSE
200', 'scheme_fund_type': u'Open-Ended', 'cat_worst_2y': u'-5.8', 'scheme_url': 'http://w
ww.moneycontrol.com/mutual-funds/nav/sbi-magnum-multicap-fund/MSB073', 'fund_family_name'
: u'SBI Mutual Fund', 'scheme_aum': u'549.09', 'cat_avg_ret_5y': u'5.7', 'scheme_name': u
'SBI Magnum Multicap Fund (G)', 'scheme_option': u'Growth', 'diff_fund_cat_3y': u'11.9',
'cat_avg_ret_1m': u'5.2', 'diff_fund_cat_3m': u'0.4', 'cat_avg_ret_1y': u'-7.7', 'categor
y': u'Diversified Equity', 'cat_worst_3m': u'-12.4', 'fund_ret_3m': u'-3.7', 'cat_avg_ret
_6m': u'-5.1', 'cat_worst_3y': u'-1.2', 'fund_ret_5y': u'12.6', 'cat_best_3y': u'30.9', '
fund_ret_1m': u'9.6', 'cat_best_2y': u'31.9', 'cat_best_3m': u'1.2', 'crisil_rating': u'R
ank 2', 'fund_ret_1y': u'-0.1'}

901 902 903 904 905 906 907 908 909 910 911 912 913 914 915
916 917 918 919 920 921 922 923 924 925 926 927 928 929 930
931 932 933 934 935 936 937 938 939 940 941 942 943 944 945
946 947 948 949 950 951 952 953 954 955 956 957 958 959 960
961 962 963 964 965 966 967 968 969 970 971 972 973 974 975
976 977 978 979 980 981 982 983 984 985 986 987 988 989 990
991 992 993 994 995 996 997 998 999 Scheme # 1000
{'fund_family_aum': u'70,780', 'fund_family_shortcode': u'TE', 'scheme_plan': u'Regular',
'scheme_risk_text': u'HIGH', 'fund_family_url': u'http://www.moneycontrol.com/mutual-fund
s/amc-details/TE', 'cat_best_6m': u'9.7', 'diff_fund_cat_1y': u'-4.4', 'cat_avg_ret_3y':
u'7.1', 'cat_avg_ret_3m': u'-0.3', 'diff_fund_cat_1m': u'0.6', 'diff_fund_cat_5y': u'None
', 'cat_worst_1m': u'-3.0', 'cat_worst_1y': u'-11.0', 'latest_nav': u'8.59', 'scheme_last
_dividend': u'None', 'scheme_min_investment': u'Rs.5000', 'diff_fund_cat_6m': u'-0.7', 'f
und_ret_3y': u'None', 'fund_ret_2y': u'None', 'cat_worst_5y': u'5.0', 'cat_best_5y': u'10
.3', '1yr_return': u'-3.8', 'category_url': 'http://www.moneycontrol.com/mutual-funds/per
formance-tracker/returns/fund-of-funds-hybrid-oriented.html', 'cat_best_1m': u'6.7', 'dif
f_fund_cat_2y': u'None', 'cat_avg_ret_2y': u'8.0', 'fund_ret_6m': u'0.5', 'cat_best_1y':
u'8.6', 'cat_worst_6m': u'-4.9', 'scheme_bonus': u'None', 'scheme_benchmark': u'N.A', 'sc
heme_fund_type': u'Open-Ended', 'cat_worst_2y': u'-17.6', 'scheme_url': 'http://www.money
control.com/mutual-funds/nav/franklin-india-feeder-franklin-european-growth-fund/MTE394',
'fund_family_name': u'Franklin Templeton Mutual Fund', 'scheme_aum': u'40.13', 'cat_avg_r
et_5y': u'4.7', 'scheme_name': u'Franklin Feeder-Franklin Euro Gr (G)', 'scheme_option':
u'Growth', 'diff_fund_cat_3y': u'None', 'cat_avg_ret_1m': u'3.1', 'diff_fund_cat_3m': u'0
.1', 'cat_avg_ret_1y': u'0.6', 'category': u'Fund of Funds - Hybrid oriented', 'cat_worst
_3m': u'-4.0', 'fund_ret_3m': u'-0.2', 'cat_avg_ret_6m': u'1.2', 'cat_worst_3y': u'-8.7',
'fund_ret_5y': u'None', 'cat_best_3y': u'15.5', 'fund_ret_1m': u'3.7', 'cat_best_2y': u'1
6.7', 'cat_best_3m': u'3.4', 'crisil_rating': u'Not Ranked', 'fund_ret_1y': u'-3.8'}

1001 1002 1003 1004 1005 1006 1007 1008 1009 1010 1011 1012 1013
1014 1015 1016 1017 1018 1019 1020 1021 1022 1023 1024 1025 1026
1027 1028 1029 1030 1031 1032 1033 1034 1035 1036 1037 1038 1039
1040 1041 1042 1043 1044 1045 1046 1047 1048 1049 1050 1051 1052
1053 1054 1055 1056 1057 1058 1059 1060 1061 1062 1063 1064 1065
1066 1067 1068 1069 1070 1071 1072 1073 1074 1075 1076 1077 1078
1079 1080 1081 1082 1083 1084 1085 1086 1087 1088 1089 1090 1091
1092 1093 1094 1095 1096 1097 1098 1099 Scheme # 1100
{'fund_family_aum': u'54,902', 'fund_family_shortcode': u'KM', 'scheme_plan': u'Regular',
'scheme_risk_text': u'LOW', 'fund_family_url': u'http://www.moneycontrol.com/mutual-funds
/amc-details/KM', 'cat_best_6m': u'4.1', 'diff_fund_cat_1y': u'1.3', 'cat_avg_ret_3y': u'
3.1', 'cat_avg_ret_3m': u'0.7', 'diff_fund_cat_1m': u'None', 'diff_fund_cat_5y': u'4.1',
'cat_worst_1m': u'0.3', 'cat_worst_1y': u'-24.7', 'latest_nav': u'18.25', 'scheme_last_di
vidend': u'None', 'scheme_min_investment': u'Rs.5000', 'diff_fund_cat_6m': u'-0.4', 'fund
_ret_3y': u'6.5', 'fund_ret_2y': u'5.6', 'cat_worst_5y': u'2.0', 'cat_best_5y': u'9.3', '
1yr_return': u'3.3', 'category_url': u'None', 'cat_best_1m': u'0.6', 'diff_fund_cat_2y':

u'2.6', 'cat_avg_ret_2y': u'3.0', 'fund_ret_6m': u'1.1', 'cat_best_1y': u'8.6', 'cat_wors
t_6m': u'0.4', 'scheme_bonus': u'None', 'scheme_benchmark': u'CRISIL Liquid Fund', 'schem
e_fund_type': u'Open-Ended', 'cat_worst_2y': u'-9.2', 'scheme_url': 'http://www.moneycont
rol.com/mutual-funds/nav/kotak-quarterly-interval-plan-series-i/MKM172', 'fund_family_nam
e': u'Kotak Mahindra Mutual Fund', 'scheme_aum': u'0.23', 'cat_avg_ret_5y': u'3.6', 'sche
me_name': u'Kotak Qtrly Interval -Sr I (G)', 'scheme_option': u'Growth', 'diff_fund_cat_3
y': u'3.4', 'cat_avg_ret_1m': u'0.2', 'diff_fund_cat_3m': u'-0.6', 'cat_avg_ret_1y': u'2.
0', 'category': u'None', 'cat_worst_3m': u'0.1', 'fund_ret_3m': u'0.1', 'cat_avg_ret_6m':
u'1.5', 'cat_worst_3y': u'-3.3', 'fund_ret_5y': u'7.7', 'cat_best_3y': u'9.2', 'fund_ret_
1m': u'None', 'cat_best_2y': u'9.1', 'cat_best_3m': u'1.9', 'crisil_rating': u'Not Ranked
', 'fund_ret_1y': u'3.3'}

1101 1102 1103 1104 1105 1106 1107 1108 1109 1110 1111 1112 1113
1114 1115 1116 1117 1118 1119 1120 1121 1122 1123 1124 1125 1126
1127 1128 1129 1130 1131 1132 1133 1134 1135 1136 1137 1138 1139
1140 1141 1142 1143 1144 1145 1146 1147 1148 1149 1150 1151 1152
1153 1154 1155 1156 1157 1158 1159 1160 1161 1162 1163 1164 1165
1166 1167 1168 1169 1170 1171 1172 1173 1174 1175 1176 1177 1178
1179 1180 1181 1182 1183 1184 1185 1186 1187 1188 1189 1190 1191
1192 1193 1194 1195 1196 1197 1198 1199 Scheme # 1200

{'fund_family_aum': u'54,715', 'fund_family_shortcode': u'AG', 'scheme_plan': u'Plan B',
'scheme_risk_text': u'MODERATE', 'fund_family_url': u'http://www.moneycontrol.com/mutual-
funds/amc-details/AG', 'cat_best_6m': u'6.8', 'diff_fund_cat_1y': u'None', 'cat_avg_ret_3
y': u'7.0', 'cat_avg_ret_3m': u'2.0', 'diff_fund_cat_1m': u'None', 'diff_fund_cat_5y': u'
None', 'cat_worst_1m': u'-0.5', 'cat_worst_1y': u'1.9', 'latest_nav': u'14.43', 'scheme_l
ast_dividend': u'None', 'scheme_min_investment': u'Rs.5000', 'diff_fund_cat_6m': u'None',
'fund_ret_3y': u'None', 'fund_ret_2y': u'None', 'cat_worst_5y': u'7.5', 'cat_best_5y': u'
11.0', '1yr_return': u'None', 'category_url': 'http://www.moneycontrol.com/mutual-funds/p
erformance-tracker/returns/debt-long-term.html', 'cat_best_1m': u'3.7', 'diff_fund_cat_2y
': u'None', 'cat_avg_ret_2y': u'9.3', 'fund_ret_6m': u'None', 'cat_best_1y': u'11.9', 'ca
t_worst_6m': u'-0.6', 'scheme_bonus': u'None', 'scheme_benchmark': u'CRISIL Composite Bon
d Fund', 'scheme_fund_type': u'Open-Ended', 'cat_worst_2y': u'5.2', 'scheme_url': 'http:/
/www.moneycontrol.com/mutual-funds/nav/idfc-super-saver-income-fund-investment-plan-b-ins
titutional-plan/MAG316', 'fund_family_name': u'IDFC Mutual Fund', 'scheme_aum': u'10.84',
'cat_avg_ret_5y': u'6.3', 'scheme_name': u'IDFC SSIF-Investment Plan B (G)', 'scheme_opti
on': u'Growth', 'diff_fund_cat_3y': u'None', 'cat_avg_ret_1m': u'1.7', 'diff_fund_cat_3m'
: u'None', 'cat_avg_ret_1y': u'5.4', 'category': u'Debt Long Term', 'cat_worst_3m': u'-0.
9', 'fund_ret_3m': u'None', 'cat_avg_ret_6m': u'2.4', 'cat_worst_3y': u'5.6', 'fund_ret_5
y': u'None', 'cat_best_3y': u'12.3', 'fund_ret_1m': u'None', 'cat_best_2y': u'13.6', 'cat
_best_3m': u'7.2', 'crisil_rating': u'Not Ranked', 'fund_ret_1y': u'None'}

1201 1202 1203 1204 1205 1206 1207 1208 1209 1210 1211 1212 1213
1214 1215 1216 1217 1218 1219 1220 1221 1222 1223 1224 1225 1226
1227 1228 1229 1230 1231 1232 1233 1234 1235 1236 1237 1238 1239
1240 1241 1242 1243 1244 1245 1246 1247 1248 1249 1250 1251 1252
1253 1254 1255 1256 1257 1258 1259 1260 1261 1262 1263 1264 1265
1266 1267 1268 1269 1270 1271 1272 1273 1274 1275 1276 1277 1278
1279 1280 1281 1282 1283 1284 1285 1286 1287 1288 1289 1290 1291
1292 1293 1294 1295

In [208]:

```
#Save the collected data to text, which is then converted to a csv file.
import csv
out_path= "./fund_schemes.txt"
out_file = open(out_path, 'wb')

fieldnames = sorted(list(set(k for d in fund_schemes for k in d)))
writer = csv.DictWriter(out_file, fieldnames=fieldnames, dialect='excel')

writer.writeheader() # Assumes Python >= 2.7
for row in fund_schemes:
    writer.writerow(row)
out_file.close()
```

In [270]:

```
'''
```

```
Method to convert risk text to a numerical attribute
'''
def encode_risk(risk_text):
    # The higher the risk, the lower the score!
    risk = {
        u'HIGH'              :  1,
        u'MODERATELY HIGH'   :  2,
        u'MODERATE'          :  3,
        u'MODERATELY LOW'    :  4,
        u'LOW'               :  5
    }
    try:
        return risk[ unicode( risk_text.upper() ) ]
    except:
        return 0

'''
Method to convert numerical features that appear as strings or unicode strings into numbe
rs
'''
def to_numeric( text ):
    try:
        return float( re.sub(
            '(Rs[ ]*\.)|[^\d|.|-]|(Rank[ ]*)',
            '',
            text,
            flags = re.IGNORECASE
        ) )
    except:
        return None
```

In [326]:

```
# Processing for ML

# Initialize a list to store metrics for ris
for idx, scheme in enumerate( fund_schemes ):
    ##
    # Step 1: Convert numerical features appearing as text to numerical features
    #         1.a: Encode risk text to a numerical representation of risk.
    #              Highest risk gets the lowest score, lowest risk gets the highest scor
e
    #
    #         1.b: Convert numbers formatted with commas or currency or rating descriptio
n to just numbers
    ##

    # Convert scheme risk text to a numerical attribute
    fund_schemes[idx]['num_scheme_risk']             = encode_risk( scheme['scheme_risk_t
ext'] )

    # Convert metrics to numerical features
    fund_schemes[idx]['num_fund_family_aum']         = to_numeric( scheme['fund_family_au
m'] )

    fund_schemes[idx]['num_crisil_rating']           = to_numeric( scheme['crisil_rating'
] )

    fund_schemes[idx]['num_latest_nav']              = to_numeric( scheme['latest_nav'] )
    fund_schemes[idx]['num_1yr_return']              = to_numeric( scheme['1yr_return'] )
    fund_schemes[idx]['num_scheme_aum']              = to_numeric( scheme['scheme_aum'] )
if scheme['scheme_aum'] != u'None' else 0

    fund_schemes[idx]['num_scheme_min_investment']   = to_numeric( scheme['scheme_min_inv
estment'] )
    fund_schemes[idx]['num_scheme_last_dividend']    = to_numeric( scheme['scheme_last_di
vidend'] )
    fund_schemes[idx]['num_scheme_bonus']            = to_numeric( scheme['scheme_bonus']
)

    fund_schemes[idx]['num_fund_ret_1m']             = to_numeric( scheme['fund_ret_1m']
```

```
)
    fund_schemes[idx]['num_fund_ret_3m']            = to_numeric( scheme['fund_ret_3m']
)
    fund_schemes[idx]['num_fund_ret_6m']            = to_numeric( scheme['fund_ret_6m']
)
    fund_schemes[idx]['num_fund_ret_1y']            = to_numeric( scheme['fund_ret_1y']
)
    fund_schemes[idx]['num_fund_ret_2y']            = to_numeric( scheme['fund_ret_2y']
)
    fund_schemes[idx]['num_fund_ret_3y']            = to_numeric( scheme['fund_ret_3y']
)
    fund_schemes[idx]['num_fund_ret_5y']            = to_numeric( scheme['fund_ret_5y']
)

    fund_schemes[idx]['num_cat_avg_ret_1m']         = to_numeric( scheme['cat_avg_ret_1m
'] )
    fund_schemes[idx]['num_cat_avg_ret_3m']         = to_numeric( scheme['cat_avg_ret_3m
'] )
    fund_schemes[idx]['num_cat_avg_ret_6m']         = to_numeric( scheme['cat_avg_ret_6m
'] )
    fund_schemes[idx]['num_cat_avg_ret_1y']         = to_numeric( scheme['cat_avg_ret_1y
'] )
    fund_schemes[idx]['num_cat_avg_ret_2y']         = to_numeric( scheme['cat_avg_ret_2y
'] )
    fund_schemes[idx]['num_cat_avg_ret_3y']         = to_numeric( scheme['cat_avg_ret_3y
'] )
    fund_schemes[idx]['num_cat_avg_ret_5y']         = to_numeric( scheme['cat_avg_ret_5y
'] )

    fund_schemes[idx]['num_diff_fund_cat_1m']       = to_numeric( scheme['diff_fund_cat_
1m'] )
    fund_schemes[idx]['num_diff_fund_cat_3m']       = to_numeric( scheme['diff_fund_cat_
3m'] )
    fund_schemes[idx]['num_diff_fund_cat_6m']       = to_numeric( scheme['diff_fund_cat_
6m'] )
    fund_schemes[idx]['num_diff_fund_cat_1y']       = to_numeric( scheme['diff_fund_cat_
1y'] )
    fund_schemes[idx]['num_diff_fund_cat_2y']       = to_numeric( scheme['diff_fund_cat_
2y'] )
    fund_schemes[idx]['num_diff_fund_cat_3y']       = to_numeric( scheme['diff_fund_cat_
3y'] )
    fund_schemes[idx]['num_diff_fund_cat_5y']       = to_numeric( scheme['diff_fund_cat_
5y'] )

    fund_schemes[idx]['num_cat_best_1m']            = to_numeric( scheme['cat_best_1m']
)
    fund_schemes[idx]['num_cat_best_3m']            = to_numeric( scheme['cat_best_3m']
)
    fund_schemes[idx]['num_cat_best_6m']            = to_numeric( scheme['cat_best_6m']
)
    fund_schemes[idx]['num_cat_best_1y']            = to_numeric( scheme['cat_best_1y']
)
    fund_schemes[idx]['num_cat_best_2y']            = to_numeric( scheme['cat_best_2y']
)
    fund_schemes[idx]['num_cat_best_3y']            = to_numeric( scheme['cat_best_3y']
)
    fund_schemes[idx]['num_cat_best_5y']            = to_numeric( scheme['cat_best_5y']
)

    fund_schemes[idx]['num_cat_worst_1m']           = to_numeric( scheme['cat_worst_1m']
)
    fund_schemes[idx]['num_cat_worst_3m']           = to_numeric( scheme['cat_worst_3m']
)
    fund_schemes[idx]['num_cat_worst_6m']           = to_numeric( scheme['cat_worst_6m']
)
    fund_schemes[idx]['num_cat_worst_1y']           = to_numeric( scheme['cat_worst_1y']
)
    fund_schemes[idx]['num_cat_worst_2y']           = to_numeric( scheme['cat_worst_2y']
)
    fund_schemes[idx]['num_cat_worst_3y']           = to_numeric( scheme['cat_worst_3y']
)
    fund_schemes[idx]['num_cat_worst_5y']           = to_numeric( scheme['cat_worst_5y']
```

```
)


    ##
    # Step 2: Calculate additional risk metrics - the fetched risk rating is based on MPT
Statistics
    #         which is already a sound measurement. Hence, we devise and incorporate mor
e measures
    #         such as:
    ##

    # Score between 0 and 1 based on Risk Rating which is based on MPT Statistics
    fund_schemes[idx]['cstm_mtrc_risk_rating'] = fund_schemes[idx]['num_scheme_risk'] /
5.0

    # Score between 0 and 1 based on CRISIL rating
    fund_schemes[idx]['cstm_mtrc_crisil']  = fund_schemes[idx]['num_crisil_rating'] / 5.
0 if fund_schemes[idx]['num_crisil_rating'] else 0

    # Score between 0 and 1 based on AUM allocation to the scheme compared to other schem
es in the fund family
    fund_schemes[idx]['cstm_mtrc_alloc'] = float( fund_schemes[idx]['num_scheme_aum'] )
/ ( fund_schemes[idx]['num_fund_family_aum'] - fund_schemes[idx]['num_scheme_aum'] )

    # Score between 0 and 1 based on fund performance relative to category performance
    fund_schemes[idx]['cstm_mtrc_diff_1m'] = 1 if fund_schemes[idx]['num_diff_fund_cat_1
m'] and fund_schemes[idx]['num_diff_fund_cat_1m'] > 0 else 0
    fund_schemes[idx]['cstm_mtrc_diff_3m'] = 1 if fund_schemes[idx]['num_diff_fund_cat_3
m'] and fund_schemes[idx]['num_diff_fund_cat_3m'] > 0 else 0
    fund_schemes[idx]['cstm_mtrc_diff_6m'] = 1 if fund_schemes[idx]['num_diff_fund_cat_6
m'] and fund_schemes[idx]['num_diff_fund_cat_6m'] > 0 else 0
    fund_schemes[idx]['cstm_mtrc_diff_1y'] = 1 if fund_schemes[idx]['num_diff_fund_cat_1
y'] and fund_schemes[idx]['num_diff_fund_cat_1y'] > 0 else 0
    fund_schemes[idx]['cstm_mtrc_diff_2y'] = 1 if fund_schemes[idx]['num_diff_fund_cat_2
y'] and fund_schemes[idx]['num_diff_fund_cat_2y'] > 0 else 0
    fund_schemes[idx]['cstm_mtrc_diff_3y'] = 1 if fund_schemes[idx]['num_diff_fund_cat_3
y'] and fund_schemes[idx]['num_diff_fund_cat_3y'] > 0 else 0
    fund_schemes[idx]['cstm_mtrc_diff_5y'] = 1 if fund_schemes[idx]['num_diff_fund_cat_5
y'] and fund_schemes[idx]['num_diff_fund_cat_5y'] > 0 else 0

    # Score between 0 and 1 based on volatility in fund's category
    fund_schemes[idx]['cstm_mtrc_volat_1m'] = float( fund_schemes[idx]['num_cat_worst_1m
'] ) / fund_schemes[idx]['num_cat_best_1m'] if fund_schemes[idx]['num_cat_worst_1m'] and
fund_schemes[idx]['num_cat_worst_1m'] >= 0 else 0
    fund_schemes[idx]['cstm_mtrc_volat_3m'] = float( fund_schemes[idx]['num_cat_worst_3m
'] ) / fund_schemes[idx]['num_cat_best_3m'] if fund_schemes[idx]['num_cat_worst_3m'] and
fund_schemes[idx]['num_cat_worst_3m'] >= 0 else 0
    fund_schemes[idx]['cstm_mtrc_volat_6m'] = float( fund_schemes[idx]['num_cat_worst_6m
'] ) / fund_schemes[idx]['num_cat_best_6m'] if fund_schemes[idx]['num_cat_worst_6m'] and
fund_schemes[idx]['num_cat_worst_6m'] >= 0 else 0
    fund_schemes[idx]['cstm_mtrc_volat_1y'] = float( fund_schemes[idx]['num_cat_worst_1y
'] ) / fund_schemes[idx]['num_cat_best_1y'] if fund_schemes[idx]['num_cat_worst_1y'] and
fund_schemes[idx]['num_cat_worst_1y'] >= 0 else 0
    fund_schemes[idx]['cstm_mtrc_volat_2y'] = float( fund_schemes[idx]['num_cat_worst_2y
'] ) / fund_schemes[idx]['num_cat_best_2y'] if fund_schemes[idx]['num_cat_worst_2y'] and
fund_schemes[idx]['num_cat_worst_2y'] >= 0 else 0
    fund_schemes[idx]['cstm_mtrc_volat_3y'] = float( fund_schemes[idx]['num_cat_worst_3y
'] ) / fund_schemes[idx]['num_cat_best_3y'] if fund_schemes[idx]['num_cat_worst_3y'] and
fund_schemes[idx]['num_cat_worst_3y'] >= 0 else 0
    fund_schemes[idx]['cstm_mtrc_volat_5y'] = float( fund_schemes[idx]['num_cat_worst_5y
'] ) / fund_schemes[idx]['num_cat_best_5y'] if fund_schemes[idx]['num_cat_worst_5y'] and
fund_schemes[idx]['num_cat_worst_5y'] >= 0 else 0



    # Initialize a set of lists to contain class labels based on time frame
    normal_scores_1m = [
        fund_schemes[idx]['cstm_mtrc_risk_rating'],
        fund_schemes[idx]['cstm_mtrc_crisil'],
        fund_schemes[idx]['cstm_mtrc_alloc'],
        fund_schemes[idx]['cstm_mtrc_diff_1m'],
        fund_schemes[idx]['cstm_mtrc_volat_1m']
```

```
    ]

    normal_scores_3m = [
        fund_schemes[idx]['cstm_mtrc_risk_rating'],
        fund_schemes[idx]['cstm_mtrc_crisil'],
        fund_schemes[idx]['cstm_mtrc_alloc'],
        fund_schemes[idx]['cstm_mtrc_diff_3m'],
        fund_schemes[idx]['cstm_mtrc_volat_3m']
    ]

    normal_scores_6m = [
        fund_schemes[idx]['cstm_mtrc_risk_rating'],
        fund_schemes[idx]['cstm_mtrc_crisil'],
        fund_schemes[idx]['cstm_mtrc_alloc'],
        fund_schemes[idx]['cstm_mtrc_diff_6m'],
        fund_schemes[idx]['cstm_mtrc_volat_6m']
    ]

    normal_scores_1y = [
        fund_schemes[idx]['cstm_mtrc_risk_rating'],
        fund_schemes[idx]['cstm_mtrc_crisil'],
        fund_schemes[idx]['cstm_mtrc_alloc'],
        fund_schemes[idx]['cstm_mtrc_diff_1y'],
        fund_schemes[idx]['cstm_mtrc_volat_1y']
    ]

    normal_scores_2y = [
        fund_schemes[idx]['cstm_mtrc_risk_rating'],
        fund_schemes[idx]['cstm_mtrc_crisil'],
        fund_schemes[idx]['cstm_mtrc_alloc'],
        fund_schemes[idx]['cstm_mtrc_diff_2y'],
        fund_schemes[idx]['cstm_mtrc_volat_2y']
    ]

    normal_scores_3y = [
        fund_schemes[idx]['cstm_mtrc_risk_rating'],
        fund_schemes[idx]['cstm_mtrc_crisil'],
        fund_schemes[idx]['cstm_mtrc_alloc'],
        fund_schemes[idx]['cstm_mtrc_diff_3y'],
        fund_schemes[idx]['cstm_mtrc_volat_3y']
    ]

    normal_scores_5y = [
        fund_schemes[idx]['cstm_mtrc_risk_rating'],
        fund_schemes[idx]['cstm_mtrc_crisil'],
        fund_schemes[idx]['cstm_mtrc_alloc'],
        fund_schemes[idx]['cstm_mtrc_diff_5y'],
        fund_schemes[idx]['cstm_mtrc_volat_5y']
    ]


    ##
    # Calculate labels for each time frame based on calculated metrics
    ##
    ##
    labels_1m = round( float( sum(normal_scores_1m ) ) / max( len( normal_scores_1m ), 1
) )
    labels_3m = round( float( sum(normal_scores_3m ) ) / max( len( normal_scores_3m ), 1
) )
    labels_6m = round( float( sum(normal_scores_6m ) ) / max( len( normal_scores_6m ), 1
) )
    labels_1y = round( float( sum(normal_scores_1y ) ) / max( len( normal_scores_1y ), 1
) )
    labels_2y = round( float( sum(normal_scores_2y ) ) / max( len( normal_scores_2y ), 1
) )
    labels_3y = round( float( sum(normal_scores_3y ) ) / max( len( normal_scores_3y ), 1
) )
    labels_5y = round( float( sum(normal_scores_5y ) ) / max( len( normal_scores_5y ), 1
) )

    # Store the labels for each time frame along with scheme details
    fund_schemes[idx]['calculated_label_1m'] = labels_1m
```

```
    fund_schemes[idx]['calculated_label_3m'] = labels_3m
    fund_schemes[idx]['calculated_label_6m'] = labels_6m
    fund_schemes[idx]['calculated_label_1y'] = labels_1y
    fund_schemes[idx]['calculated_label_2y'] = labels_2y
    fund_schemes[idx]['calculated_label_3y'] = labels_3y
    fund_schemes[idx]['calculated_label_5y'] = labels_5y
```

In [328]:

```
some_val = list()
for scheme in fund_schemes:
    if scheme['num_crisil_rating'] not in some_val:
        some_val.append( scheme['num_crisil_rating'] )

print set( some_val )
```

```
set([0.0, 1.0, 2.0, 3.0, 4.0, 5.0])
```

In [ ]:

```
import csv

keys = toCSV[0].keys()
with open('funds.csv', 'wb') as output_file:
    dict_writer = csv.DictWriter(output_file, keys)
    dict_writer.writeheader()
    dict_writer.writerows(toCSV)
```

In [307]:

```
##
# Create target values for each time frame
##
Y_1m = np.array( [scheme['calculated_label_1m'] for scheme in fund_schemes] )
Y_3m = np.array( [scheme['calculated_label_3m'] for scheme in fund_schemes] )
Y_6m = np.array( [scheme['calculated_label_6m'] for scheme in fund_schemes] )
Y_1y = np.array( [scheme['calculated_label_1y'] for scheme in fund_schemes] )
Y_2y = np.array( [scheme['calculated_label_2y'] for scheme in fund_schemes] )
Y_3y = np.array( [scheme['calculated_label_3y'] for scheme in fund_schemes] )
Y_5y = np.array( [scheme['calculated_label_5y'] for scheme in fund_schemes] )
```

```
(array([ 0.,  0.,  0., ...,  0.,  0.,  0.]), array([ 0.,  0.,  0., ...,  0.,  0.,  0.]),
array([ 0.,  0.,  0., ...,  0.,  0.,  0.]), array([ 0.,  0.,  0., ...,  0.,  0.,  0.]), a
rray([ 0.,  0.,  0., ...,  0.,  0.,  0.]), array([ 1.,  1.,  1., ...,  0.,  0.,  0.]), ar
ray([ 0.,  1.,  1., ...,  0.,  0.,  0.]))
```

In [422]:

```
##
# Create feature vectors for each time frame
##
X_1m = np.array(
    [
        [
            scheme['num_scheme_risk']           if scheme['num_scheme_risk'] else 0,
            scheme['num_crisil_rating']         if scheme['num_crisil_rating'] else 0,
            scheme['num_fund_family_aum']       if scheme['num_fund_family_aum'] else 0,
            scheme['num_scheme_aum']            if scheme['num_scheme_aum'] else 0,
            scheme['num_latest_nav']            if scheme['num_latest_nav'] else 0,
            scheme['num_scheme_min_investment'] if scheme['num_scheme_min_investment'] e
lse 0,
            scheme['num_scheme_last_dividend']  if scheme['num_scheme_last_dividend'] el
se 0,
            scheme['num_scheme_bonus']          if scheme['num_scheme_bonus'] else 0,
            scheme['num_fund_ret_1m']           if scheme['num_fund_ret_1m'] else 0,
            scheme['num_cat_avg_ret_1m']        if scheme['num_cat_avg_ret_1m'] else 0
        ]
        for scheme in fund_schemes
    ]
)
```

```
X_3m = np.array(
    [
        [
            scheme['num_scheme_risk']           if scheme['num_scheme_risk'] else 0,
            scheme['num_crisil_rating']         if scheme['num_crisil_rating'] else 0,
            scheme['num_fund_family_aum']       if scheme['num_fund_family_aum'] else 0,
            scheme['num_scheme_aum']            if scheme['num_scheme_aum'] else 0,
            scheme['num_latest_nav']            if scheme['num_latest_nav'] else 0,
            scheme['num_scheme_min_investment'] if scheme['num_scheme_min_investment'] e
lse 0,
            scheme['num_scheme_last_dividend']  if scheme['num_scheme_last_dividend'] el
se 0,
            scheme['num_scheme_bonus']          if scheme['num_scheme_bonus'] else 0,
            scheme['num_fund_ret_1m']           if scheme['num_fund_ret_3m'] else 0,
            scheme['num_cat_avg_ret_1m']        if scheme['num_cat_avg_ret_3m'] else 0
        ]
        for scheme in fund_schemes
    ], dtype = 'float64'
)

X_6m = np.array(
    [
        [
            scheme['num_scheme_risk']           if scheme['num_scheme_risk'] else 0,
            scheme['num_crisil_rating']         if scheme['num_crisil_rating'] else 0,
            scheme['num_fund_family_aum']       if scheme['num_fund_family_aum'] else 0,
            scheme['num_scheme_aum']            if scheme['num_scheme_aum'] else 0,
            scheme['num_latest_nav']            if scheme['num_latest_nav'] else 0,
            scheme['num_scheme_min_investment'] if scheme['num_scheme_min_investment'] e
lse 0,
            scheme['num_scheme_last_dividend']  if scheme['num_scheme_last_dividend'] el
se 0,
            scheme['num_scheme_bonus']          if scheme['num_scheme_bonus'] else 0,
            scheme['num_fund_ret_1m']           if scheme['num_fund_ret_6m'] else 0,
            scheme['num_cat_avg_ret_1m']        if scheme['num_cat_avg_ret_6m'] else 0
        ]
        for scheme in fund_schemes
    ], dtype = 'float64'
)

X_1y = np.array(
    [
        [
            scheme['num_scheme_risk']           if scheme['num_scheme_risk'] else 0,
            scheme['num_crisil_rating']         if scheme['num_crisil_rating'] else 0,
            scheme['num_fund_family_aum']       if scheme['num_fund_family_aum'] else 0,
            scheme['num_scheme_aum']            if scheme['num_scheme_aum'] else 0,
            scheme['num_latest_nav']            if scheme['num_latest_nav'] else 0,
            scheme['num_scheme_min_investment'] if scheme['num_scheme_min_investment'] e
lse 0,
            scheme['num_scheme_last_dividend']  if scheme['num_scheme_last_dividend'] el
se 0,
            scheme['num_scheme_bonus']          if scheme['num_scheme_bonus'] else 0,
            scheme['num_fund_ret_1m']           if scheme['num_fund_ret_1y'] else 0,
            scheme['num_cat_avg_ret_1m']        if scheme['num_cat_avg_ret_1y'] else 0
        ]
        for scheme in fund_schemes
    ], dtype = 'float64'
)

X_2y = np.array(
    [
        [
            scheme['num_scheme_risk']           if scheme['num_scheme_risk'] else 0,
            scheme['num_crisil_rating']         if scheme['num_crisil_rating'] else 0,
            scheme['num_fund_family_aum']       if scheme['num_fund_family_aum'] else 0,
            scheme['num_scheme_aum']            if scheme['num_scheme_aum'] else 0,
            scheme['num_latest_nav']            if scheme['num_latest_nav'] else 0,
            scheme['num_scheme_min_investment'] if scheme['num_scheme_min_investment'] e
lse 0,
            scheme['num_scheme_last_dividend']  if scheme['num_scheme_last_dividend'] el
se 0,
```

```
            scheme['num_scheme_bonus']           if scheme['num_scheme_bonus'] else 0,
            scheme['num_fund_ret_1m']            if scheme['num_fund_ret_2y'] else 0,
            scheme['num_cat_avg_ret_1m']         if scheme['num_cat_avg_ret_2y'] else 0
        ]
        for scheme in fund_schemes
    ], dtype = 'float64'
)

X_3y = np.array(
    [
        [
            scheme['num_scheme_risk']            if scheme['num_scheme_risk'] else 0,
            scheme['num_crisil_rating']          if scheme['num_crisil_rating'] else 0,
            scheme['num_fund_family_aum']        if scheme['num_fund_family_aum'] else 0,
            scheme['num_scheme_aum']             if scheme['num_scheme_aum'] else 0,
            scheme['num_latest_nav']             if scheme['num_latest_nav'] else 0,
            scheme['num_scheme_min_investment'] if scheme['num_scheme_min_investment'] e
lse 0,
            scheme['num_scheme_last_dividend']  if scheme['num_scheme_last_dividend'] el
se 0,
            scheme['num_scheme_bonus']           if scheme['num_scheme_bonus'] else 0,
            scheme['num_fund_ret_1m']            if scheme['num_fund_ret_3y'] else 0,
            scheme['num_cat_avg_ret_1m']         if scheme['num_cat_avg_ret_3y'] else 0
        ]
        for scheme in fund_schemes
    ], dtype = 'float64'
)

X_5y = np.array(
    [
        [
            scheme['num_scheme_risk']            if scheme['num_scheme_risk'] else 0,
            scheme['num_crisil_rating']          if scheme['num_crisil_rating'] else 0,
            scheme['num_fund_family_aum']        if scheme['num_fund_family_aum'] else 0,
            scheme['num_scheme_aum']             if scheme['num_scheme_aum'] else 0,
            scheme['num_latest_nav']             if scheme['num_latest_nav'] else 0,
            scheme['num_scheme_min_investment'] if scheme['num_scheme_min_investment'] e
lse 0,
            scheme['num_scheme_last_dividend']  if scheme['num_scheme_last_dividend'] el
se 0,
            scheme['num_scheme_bonus']           if scheme['num_scheme_bonus'] else 0,
            scheme['num_fund_ret_1m']            if scheme['num_fund_ret_5y'] else 0,
            scheme['num_cat_avg_ret_1m']         if scheme['num_cat_avg_ret_5y'] else 0
        ]
        for scheme in fund_schemes
    ], dtype = 'float64'
)



# Handle NaNs using an Imputer
from sklearn.preprocessing import Imputer

imp = Imputer(missing_values='NaN', strategy='mean', axis=0)

X_1m = imp.fit_transform( X_1m )
X_3m = imp.fit_transform( X_3m )
X_6m = imp.fit_transform( X_6m )
X_1y = imp.fit_transform( X_1y )
X_2y = imp.fit_transform( X_2y )
X_3y = imp.fit_transform( X_3y )
X_5y = imp.fit_transform( X_5y )
```

In [487]:

```
# Use Random forest classifer and cross validation for number of trees ranging from 1 to
30
# to find out which trees gives more accuracy.

num_trees = range(1, 41)
```

```
# Define folds = N for N-fold cross-validation
num_folds = 10

# Define a DF to store cross validation results
df_rf_1m = pd.DataFrame()
df_rf_3m = pd.DataFrame()
df_rf_6m = pd.DataFrame()
df_rf_1y = pd.DataFrame()
df_rf_2y = pd.DataFrame()
df_rf_3y = pd.DataFrame()
df_rf_5y = pd.DataFrame()

df_rf_1m['num_trees'] = [0]  * len( num_trees )
df_rf_1m['scores']    = [[]] * len( num_trees )

df_rf_3m['num_trees'] = [0]  * len( num_trees )
df_rf_3m['scores']    = [[]] * len( num_trees )

df_rf_6m['num_trees'] = [0]  * len( num_trees )
df_rf_6m['scores']    = [[]] * len( num_trees )

df_rf_1y['num_trees'] = [0]  * len( num_trees )
df_rf_1y['scores']    = [[]] * len( num_trees )

df_rf_2y['num_trees'] = [0]  * len( num_trees )
df_rf_2y['scores']    = [[]] * len( num_trees )

df_rf_3y['num_trees'] = [0]  * len( num_trees )
df_rf_3y['scores']    = [[]] * len( num_trees )

df_rf_5y['num_trees'] = [0]  * len( num_trees )
df_rf_5y['scores']    = [[]] * len( num_trees )

# compute score for various number of trees using RandomForestClassifier for each time fr
ame.
for num in num_trees:
    forest = sklearn.ensemble.RandomForestClassifier(n_estimators = num)

    scores_1m = sklearn.cross_validation.cross_val_score(forest, X_1m[:1000, :], Y_1m[:1
000], scoring = 'f1', cv = num_folds)
    scores_3m = sklearn.cross_validation.cross_val_score(forest, X_3m[:1000, :], Y_3m[:1
000], scoring = 'f1', cv = num_folds)
    scores_6m = sklearn.cross_validation.cross_val_score(forest, X_6m[:1000, :], Y_6m[:1
000], scoring = 'f1', cv = num_folds)
    scores_1y = sklearn.cross_validation.cross_val_score(forest, X_1y[:1000, :], Y_1y[:1
000], scoring = 'f1', cv = num_folds)
    scores_2y = sklearn.cross_validation.cross_val_score(forest, X_2y[:1000, :], Y_2y[:1
000], scoring = 'f1', cv = num_folds)
    scores_3y = sklearn.cross_validation.cross_val_score(forest, X_3y[:1000, :], Y_3y[:1
000], scoring = 'f1', cv = num_folds)
    scores_5y = sklearn.cross_validation.cross_val_score(forest, X_5y[:1000, :], Y_5y[:1
000], scoring = 'f1', cv = num_folds)

    df_rf_1m['num_trees'][ num - 1] = num
    df_rf_3m['num_trees'][ num - 1] = num
    df_rf_6m['num_trees'][ num - 1] = num
    df_rf_1y['num_trees'][ num - 1] = num
    df_rf_2y['num_trees'][ num - 1] = num
    df_rf_3y['num_trees'][ num - 1] = num
    df_rf_5y['num_trees'][ num - 1] = num

    df_rf_1m['scores'][ num - 1] = scores_1m
    df_rf_3m['scores'][ num - 1] = scores_3m
    df_rf_6m['scores'][ num - 1] = scores_6m
    df_rf_1y['scores'][ num - 1] = scores_1y
    df_rf_2y['scores'][ num - 1] = scores_2y
    df_rf_3y['scores'][ num - 1] = scores_3y
    df_rf_5y['scores'][ num - 1] = scores_5y
```

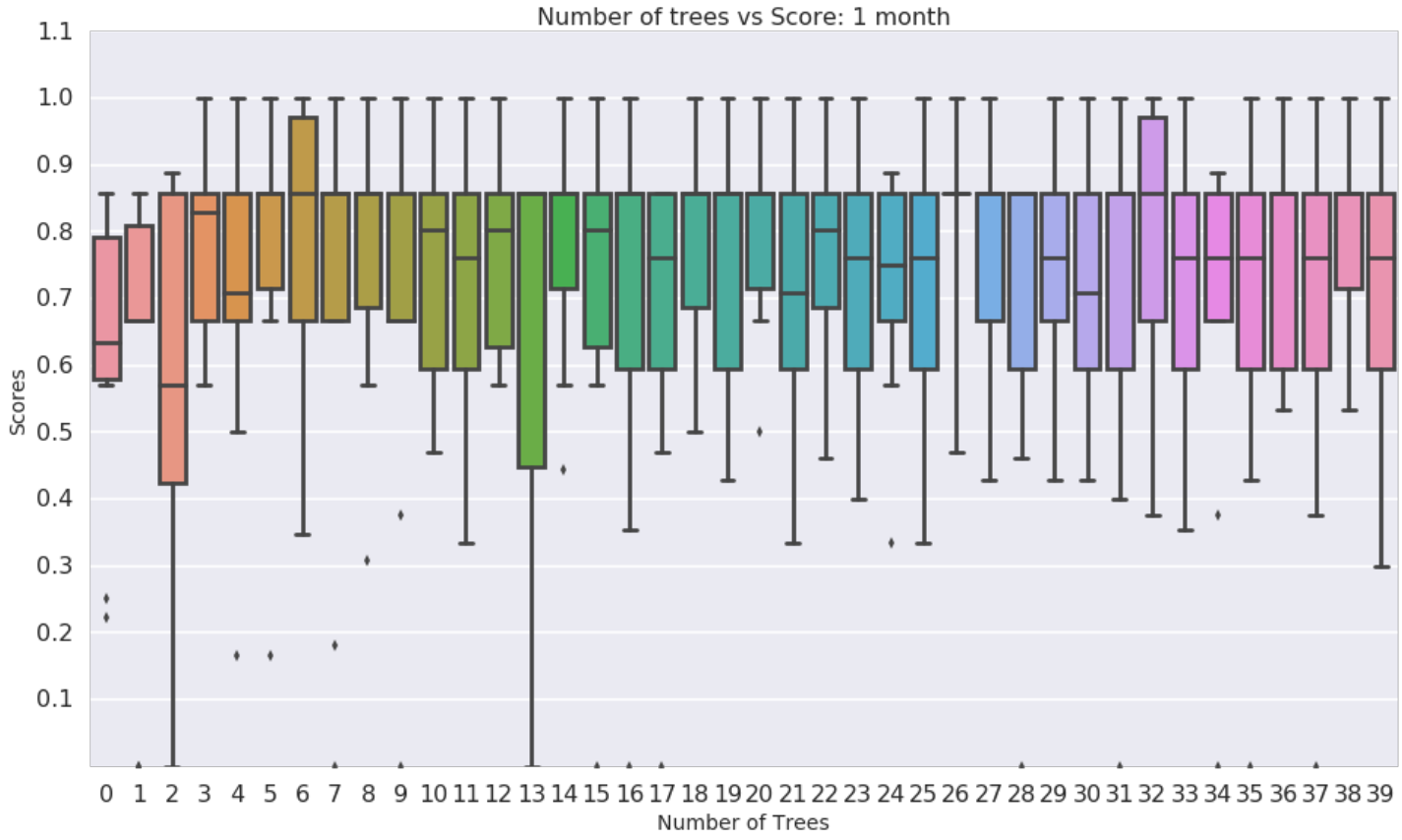
```
/home/ubuntu/anaconda2/lib/python2.7/site-packages/ipykernel/__main__.py:51: SettingWithC
opyWarning:
A value is trying to be set on a copy of a slice from a DataFrame
```

```
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexin
g.html#indexing-view-versus-copy
/home/ubuntu/anaconda2/lib/python2.7/site-packages/ipykernel/__main__.py:52: SettingWithC
opyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexin
g.html#indexing-view-versus-copy
/home/ubuntu/anaconda2/lib/python2.7/site-packages/ipykernel/__main__.py:53: SettingWithC
opyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexin
g.html#indexing-view-versus-copy
/home/ubuntu/anaconda2/lib/python2.7/site-packages/ipykernel/__main__.py:54: SettingWithC
opyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexin
g.html#indexing-view-versus-copy
/home/ubuntu/anaconda2/lib/python2.7/site-packages/ipykernel/__main__.py:55: SettingWithC
opyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexin
g.html#indexing-view-versus-copy
/home/ubuntu/anaconda2/lib/python2.7/site-packages/ipykernel/__main__.py:56: SettingWithC
opyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexin
g.html#indexing-view-versus-copy
/home/ubuntu/anaconda2/lib/python2.7/site-packages/ipykernel/__main__.py:57: SettingWithC
opyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexin
g.html#indexing-view-versus-copy
/home/ubuntu/anaconda2/lib/python2.7/site-packages/ipykernel/__main__.py:59: SettingWithC
opyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexin
g.html#indexing-view-versus-copy
/home/ubuntu/anaconda2/lib/python2.7/site-packages/ipykernel/__main__.py:60: SettingWithC
opyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexin
g.html#indexing-view-versus-copy
/home/ubuntu/anaconda2/lib/python2.7/site-packages/ipykernel/__main__.py:61: SettingWithC
opyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexin
g.html#indexing-view-versus-copy
/home/ubuntu/anaconda2/lib/python2.7/site-packages/ipykernel/__main__.py:62: SettingWithC
opyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexin
g.html#indexing-view-versus-copy
/home/ubuntu/anaconda2/lib/python2.7/site-packages/ipykernel/__main__.py:63: SettingWithC
opyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexin
g.html#indexing-view-versus-copy
/home/ubuntu/anaconda2/lib/python2.7/site-packages/ipykernel/__main__.py:64: SettingWithC
opyWarning:
A value is trying to be set on a copy of a slice from a DataFrame
```

```
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy
/home/ubuntu/anaconda2/lib/python2.7/site-packages/ipykernel/__main__.py:65: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy
```

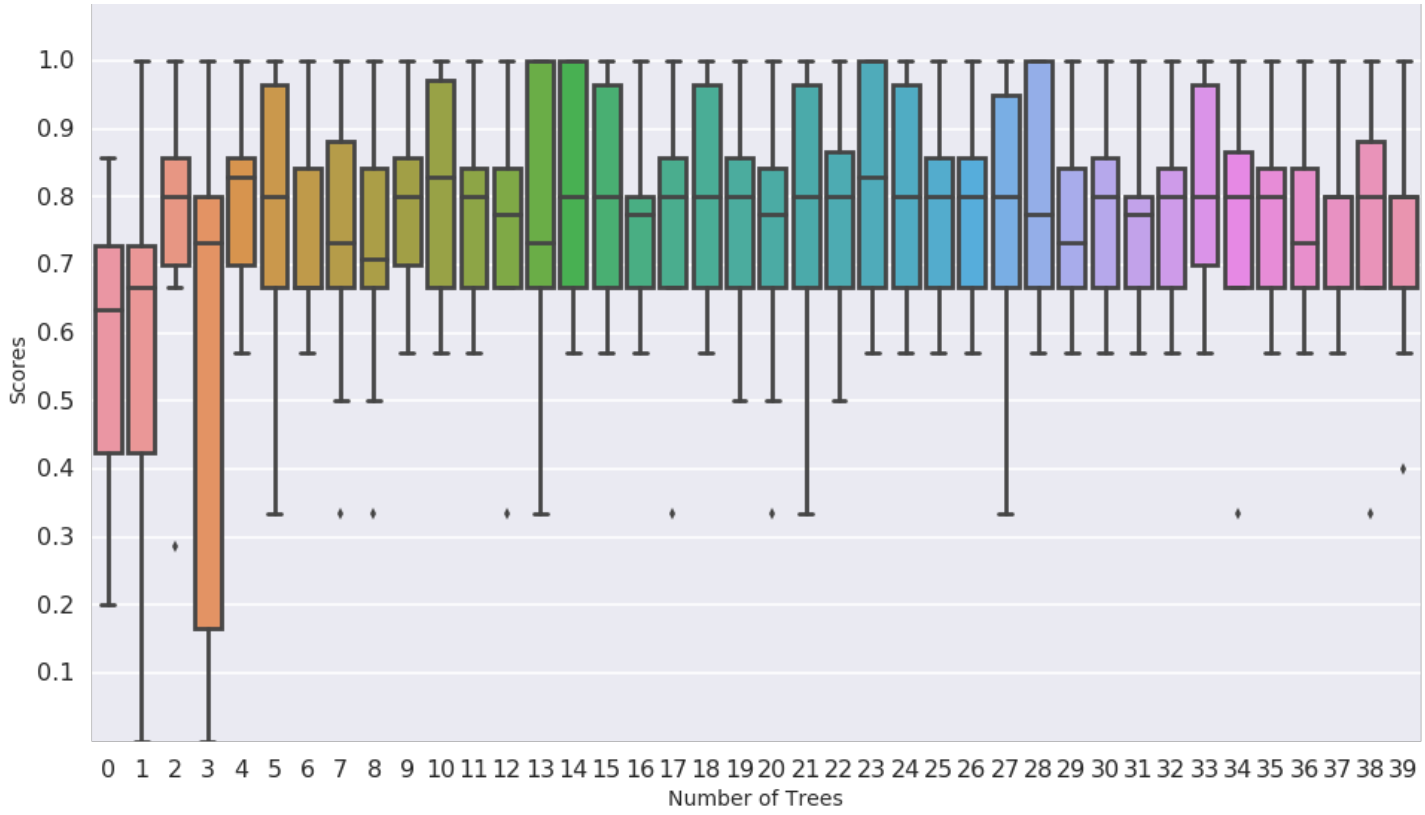
In [495]:

```
# plot the scores of the random forests as a function of the number of trees
plt.figure(figsize=(16,9))

# Scores of 10-fold cross-validation for random forests ranging from 1 to 40 trees as a box plot
sns.boxplot(data  = df_rf_1m.scores,
            names = df_rf_1m.num_trees.values )

plt.title(  "Number of trees vs Score: 1 month", fontsize=16)
plt.xlabel( "Number of Trees", fontsize=14)
plt.ylabel( "Scores", fontsize=14)
plt.yticks( np.arange( 0.1, 1.2, 0.1 ) )
sns.set_context('poster')
```



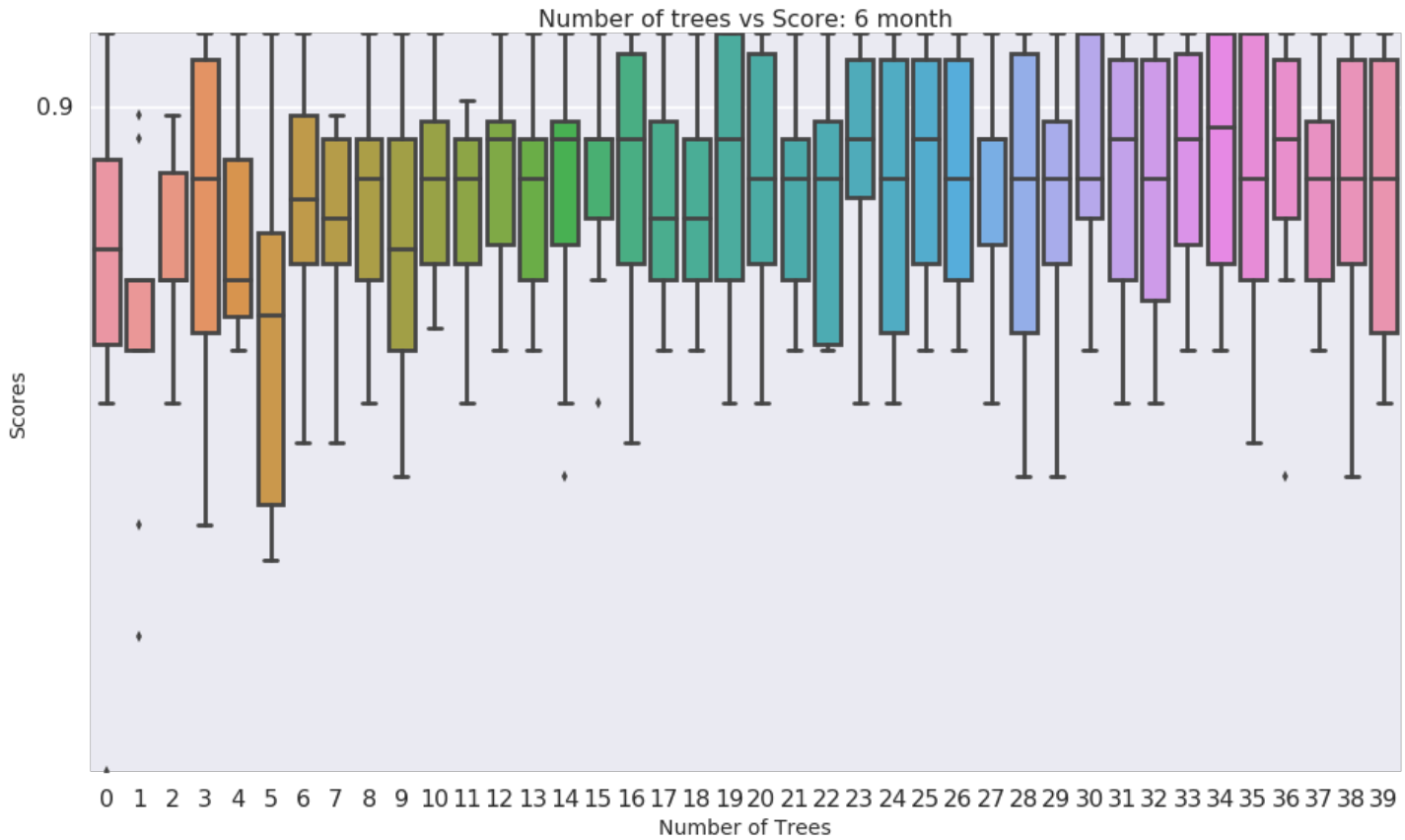
In [489]:

```
# plot the scores of the random forests as a function of the number of trees
plt.figure(figsize=(16,9))

# Scores of 10-fold cross-validation for random forests ranging from 1 to 40 trees as a box plot
sns.boxplot(data  = df_rf_3m.scores,
            names = df_rf_3m.num_trees.values )

plt.title(  "Number of trees vs Score: 3 month", fontsize=16)
plt.xlabel( "Number of Trees", fontsize=14)
plt.ylabel( "Scores", fontsize=14)
plt.yticks( np.arange( 0.1, 1.2, 0.1 ) )
sns.set_context('poster')
```



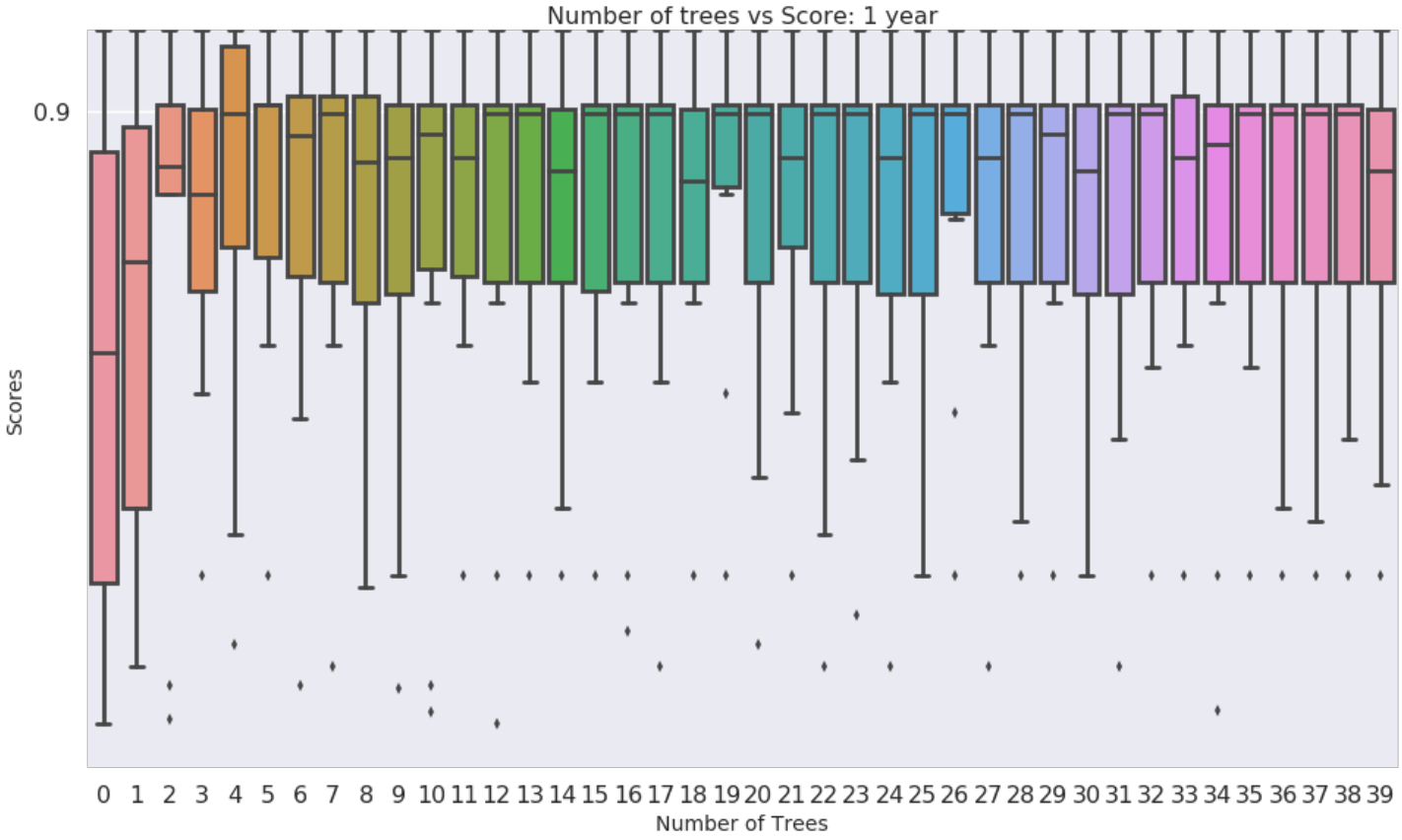
In [490]:

```
# plot the scores of the random forests as a function of the number of trees
plt.figure(figsize=(16,9))

# Scores of 10-fold cross-validation for random forests ranging from 1 to 40 trees as a b
ox plot
sns.boxplot(data  = df_rf_6m.scores,
            names = df_rf_6m.num_trees.values )

plt.title(  "Number of trees vs Score: 6 month", fontsize=16)
plt.xlabel( "Number of Trees", fontsize=14)
plt.ylabel( "Scores", fontsize=14)
plt.yticks( np.arange( 0.9, 1.0, 0.1 ) )
sns.set_context('poster')
```



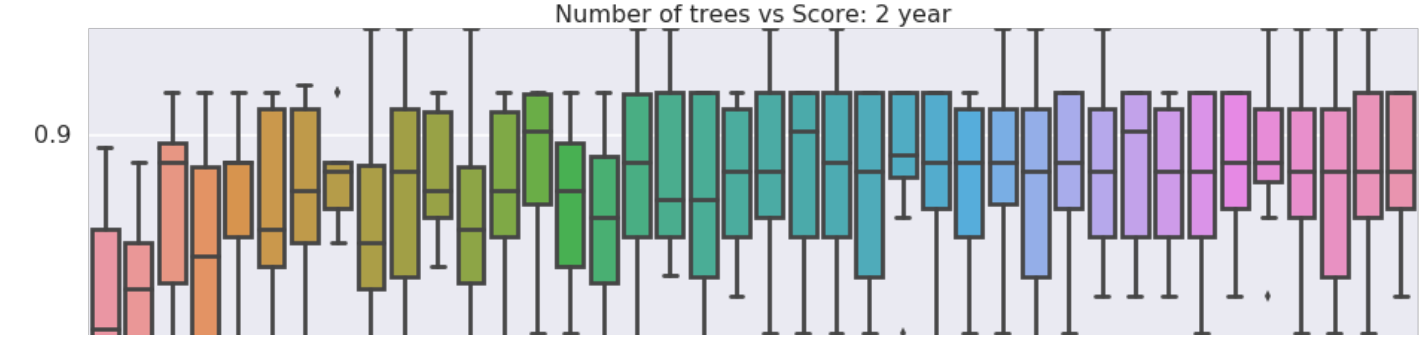
In [491]:

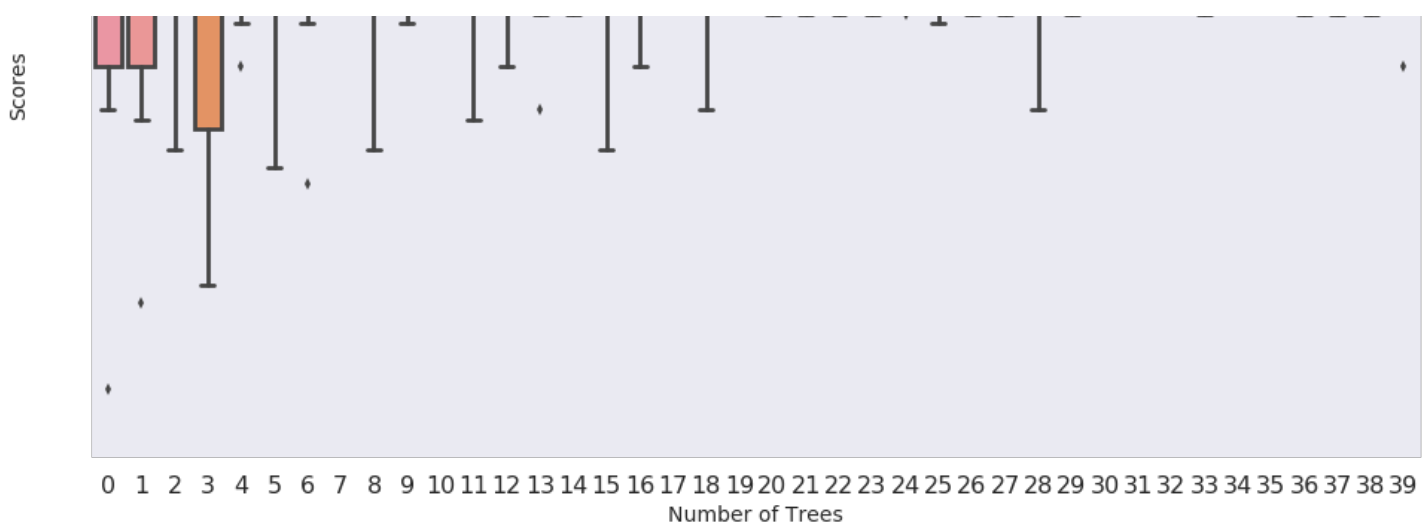
```
# plot the scores of the random forests as a function of the number of trees
plt.figure(figsize=(16,9))

# Scores of 10-fold cross-validation for random forests ranging from 1 to 40 trees as a b
ox plot
sns.boxplot(data  = df_rf_1y.scores,
            names = df_rf_1y.num_trees.values )

plt.title(  "Number of trees vs Score: 1 year", fontsize=16)
plt.xlabel( "Number of Trees", fontsize=14)
plt.ylabel( "Scores", fontsize=14)
plt.yticks( np.arange( 0.9, 1.0, 0.1 ) )
sns.set_context('poster')
```



In [492]:

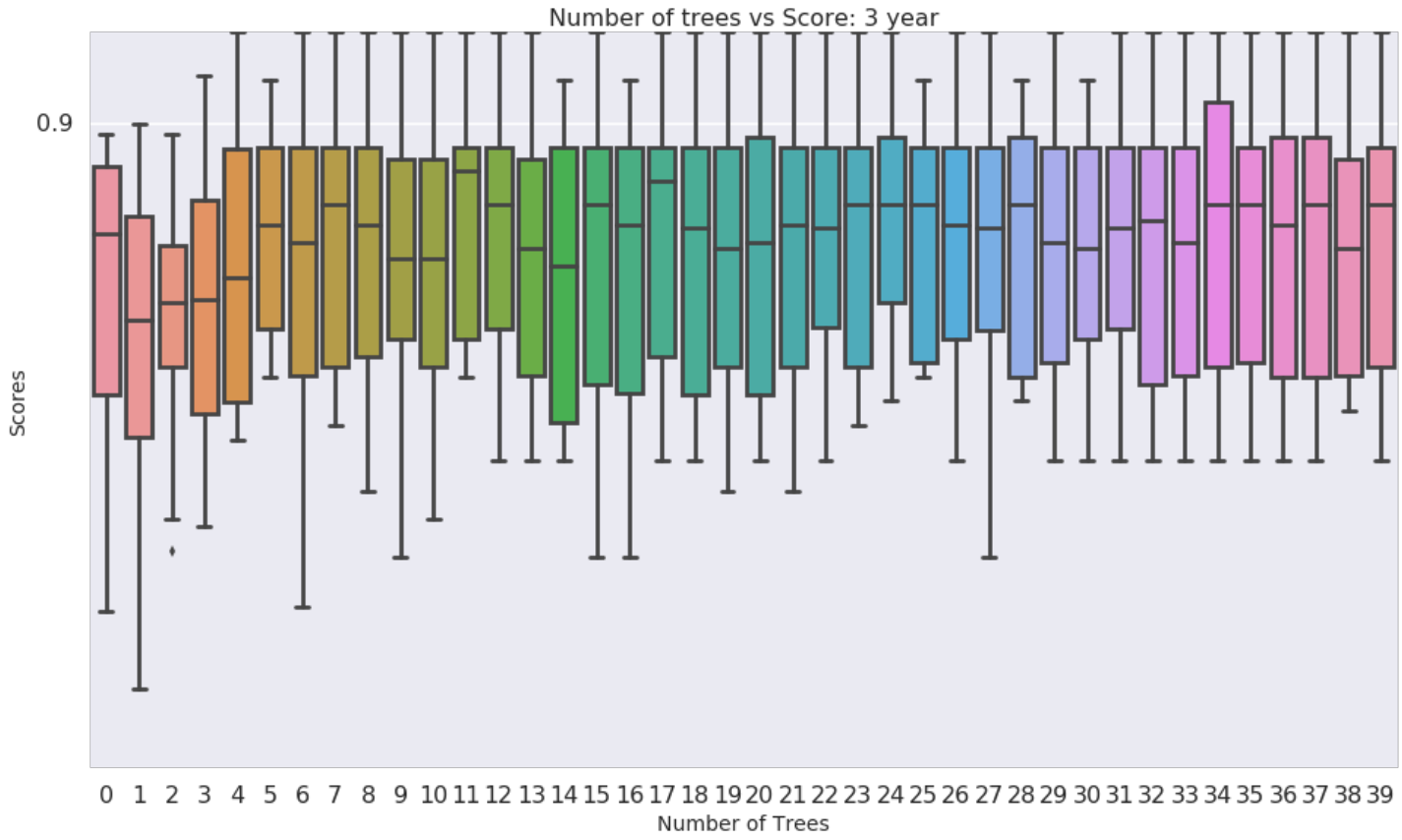
```
# plot the scores of the random forests as a function of the number of trees
plt.figure(figsize=(16,9))

# Scores of 10-fold cross-validation for random forests ranging from 1 to 40 trees as a b
ox plot
sns.boxplot(data  = df_rf_2y.scores,
            names = df_rf_2y.num_trees.values )

plt.title(  "Number of trees vs Score: 2 year", fontsize=16)
plt.xlabel( "Number of Trees", fontsize=14)
plt.ylabel( "Scores", fontsize=14)
plt.yticks( np.arange( 0.9, 1.0, 0.1 ) )
sns.set_context('poster')
```





In [493]:

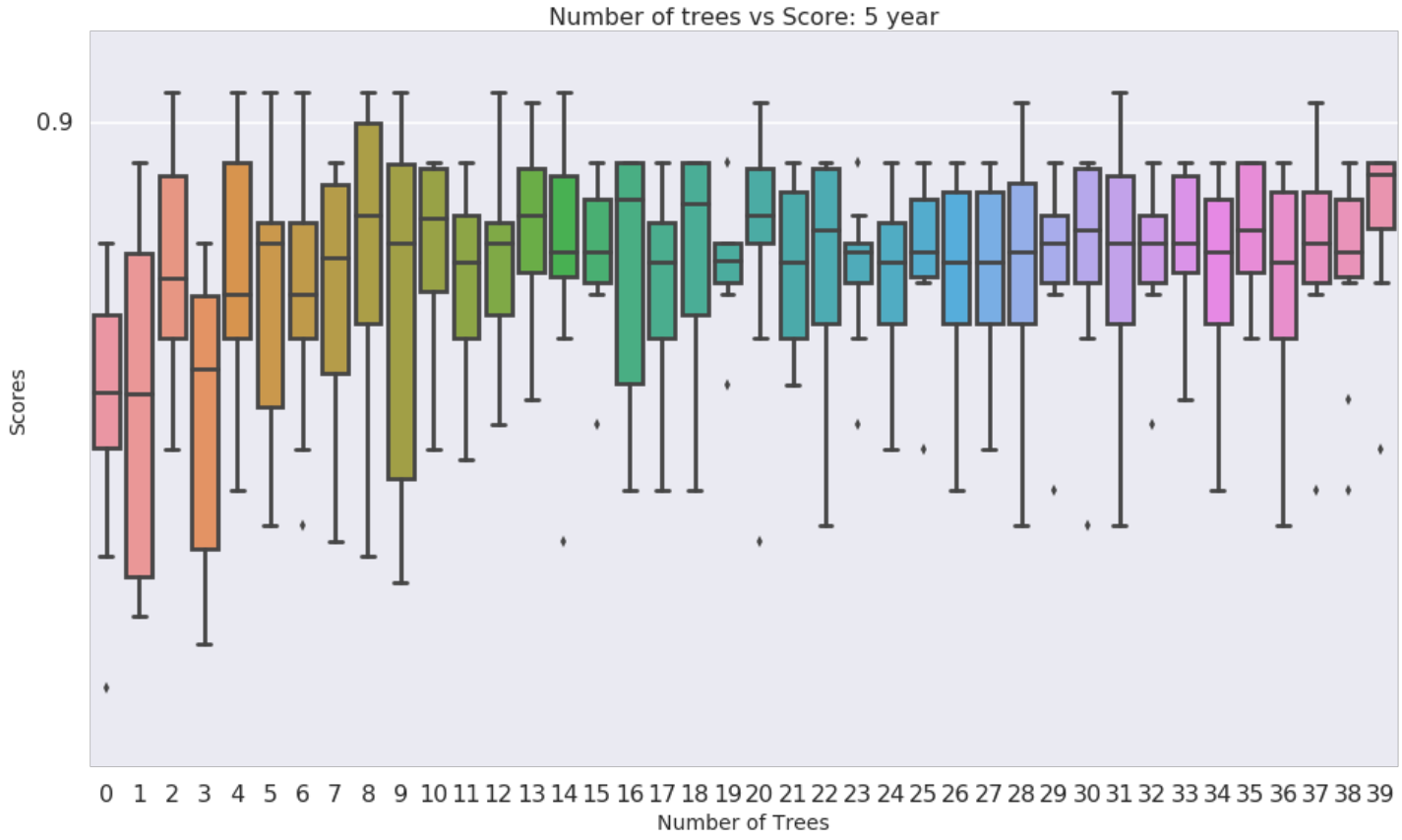
```
# plot the scores of the random forests as a function of the number of trees
plt.figure(figsize=(16,9))

# Scores of 10-fold cross-validation for random forests ranging from 1 to 40 trees as a b
ox plot
sns.boxplot(data  = df_rf_3y.scores,
            names = df_rf_3y.num_trees.values )

plt.title(  "Number of trees vs Score: 3 year", fontsize=16)
plt.xlabel( "Number of Trees", fontsize=14)
plt.ylabel( "Scores", fontsize=14)
plt.yticks( np.arange( 0.9, 1.0, 0.1 ) )
sns.set_context('poster')
```



In [494]:

```
# plot the scores of the random forests as a function of the number of trees
plt.figure(figsize=(16,9))

# Scores of 10-fold cross-validation for random forests ranging from 1 to 40 trees as a b
ox plot
sns.boxplot(data  = df_rf_5y.scores,
            names = df_rf_5y.num_trees.values )
```

```
plt.title(  "Number of trees vs Score: 5 year", fontsize=16)
plt.xlabel( "Number of Trees", fontsize=14)
plt.ylabel( "Scores", fontsize=14)
plt.yticks( np.arange( 0.9, 1.0, 0.1 ) )
sns.set_context('poster')
```



In [496]:

```
##
# 1 month:
##
# Train random forest classifier with the optimal 27 estimators
##
clf_1m = sklearn.ensemble.RandomForestClassifier( n_estimators = 27)
clf_1m = clf_1m.fit( X_1m[:1000, :], Y_1m[:1000] )

##
# 3 month:
##
# Train random forest classifier with the optimal 3 estimators
##
clf_3m = sklearn.ensemble.RandomForestClassifier( n_estimators = 3)
clf_3m = clf_3m.fit( X_3m[:1000, :], Y_3m[:1000] )

##
# 6 month:
##
# Train random forest classifier with the optimal 2 estimators
##
clf_6m = sklearn.ensemble.RandomForestClassifier( n_estimators = 2)
clf_6m = clf_6m.fit( X_6m[:1000, :], Y_6m[:1000] )

##
# 1 year:
##
# Train random forest classifier with the optimal 20 estimators
##
clf_1y = sklearn.ensemble.RandomForestClassifier( n_estimators = 20)
clf_1y = clf_1y.fit( X_1y[:1000, :], Y_1y[:1000] )

##
# 2 year:
```

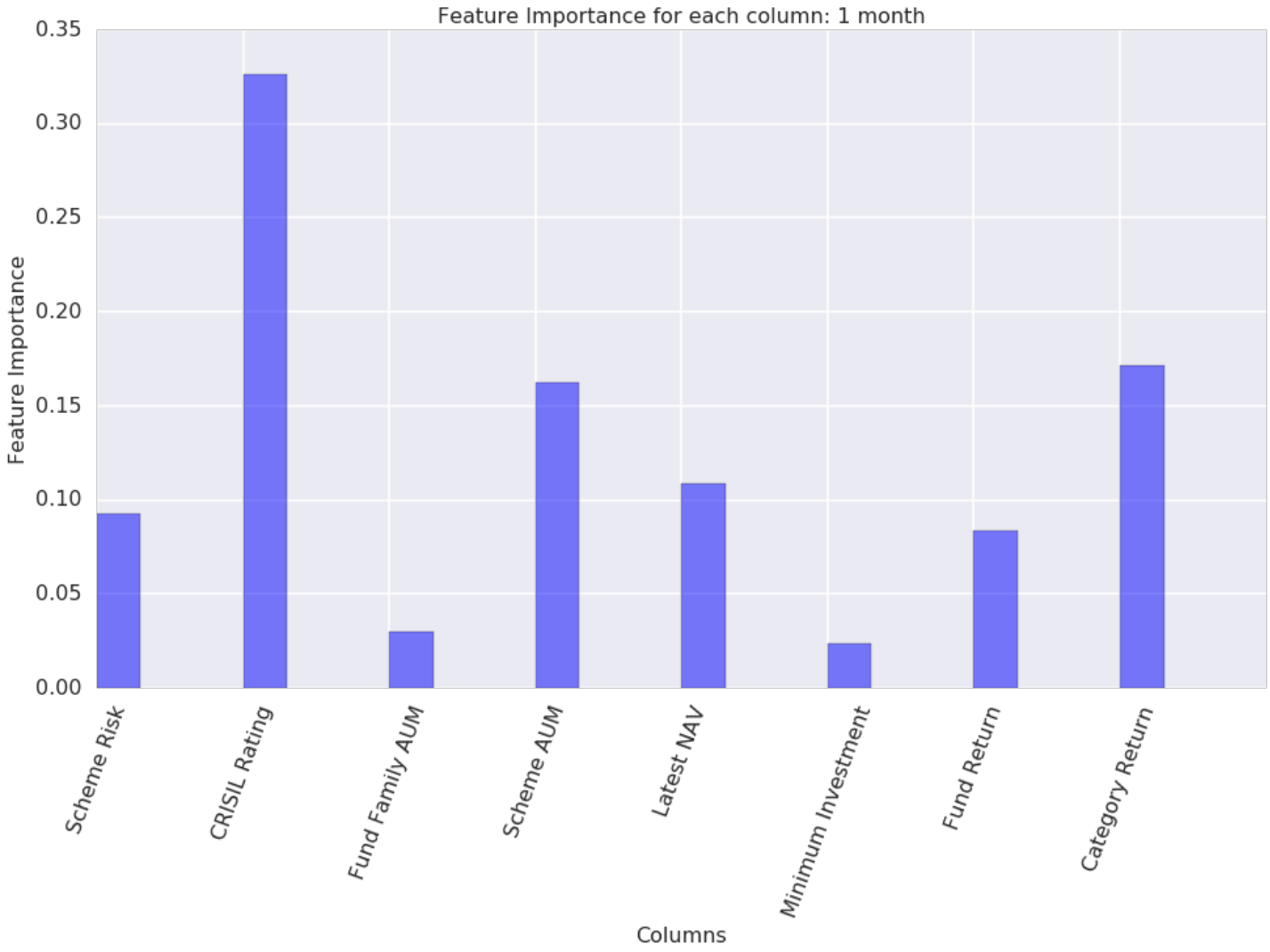
```
##
# Train random forest classifier with the optimal 8 estimators
##
clf_2y = sklearn.ensemble.RandomForestClassifier( n_estimators = 8)
clf_2y = clf_2y.fit( X_2y[:1000, :], Y_2y[:1000] )

##
# 3 year:
##
# Train random forest classifier with the optimal 3 estimators
##
clf_3y = sklearn.ensemble.RandomForestClassifier( n_estimators = 3)
clf_3y = clf_3y.fit( X_3y[:1000, :], Y_3y[:1000] )

##
# 5 year:
##
# Train random forest classifier with the optimal 20 estimators
##
clf_5y = sklearn.ensemble.RandomForestClassifier( n_estimators = 20)
clf_5y = clf_5y.fit( X_5y[:1000, :], Y_5y[:1000] )



# obtain the relative importance of the features
feature_imp_1m = clf_1m.feature_importances_
feature_imp_3m = clf_3m.feature_importances_
feature_imp_6m = clf_6m.feature_importances_
feature_imp_1y = clf_1y.feature_importances_
feature_imp_2y = clf_2y.feature_importances_
feature_imp_3y = clf_3y.feature_importances_
feature_imp_5y = clf_5y.feature_importances_

#get column names
columns = ['Scheme Risk',
           'CRISIL Rating',
           'Fund Family AUM',
           'Scheme AUM',
           'Latest NAV',
           'Minimum Investment',
           'Last Dividend',
           'Bonus',
           'Fund Return',
           'Category Return'
         ]

# Diagnostics - Check relative importance of features
print feature_imp_1m
print feature_imp_3m
print feature_imp_6m
print feature_imp_1y
print feature_imp_2y
print feature_imp_3y
print feature_imp_5y


# Plot feature importances for each time frame
index = np.arange( len(columns) - 2 )
bar_width = 0.3
opacity = 0.5
```

```
[ 0.09282479  0.32644529  0.03007116  0.16264942  0.10900816  0.02359042
  0.          0.          0.08362599  0.17178477]
[ 0.03819964  0.23473942  0.09508032  0.08693189  0.1384061   0.00641235
  0.          0.          0.06095899  0.33927129]
[ 0.05316817  0.31008474  0.02494657  0.27854677  0.04855118  0.04869794
  0.          0.          0.11342792  0.1225767 ]
[ 0.13872949  0.05029554  0.02436907  0.09845752  0.19548184  0.02868523
  0.          0.          0.21808569  0.24589563]
[ 0.09117601  0.15681827  0.04072377  0.16006965  0.22862817  0.02336717
  0.          0.          0.1487865   0.15043047]
[ 0.09450338  0.35734424  0.02171873  0.20438698  0.10816254  0.01229654
```

```
  0.          0.          0.08950784  0.11207975]
[ 0.08517348  0.26857543  0.03617007  0.191835    0.09880103  0.02094613
  0.          0.          0.16150257  0.1369963 ]
```

In [497]:

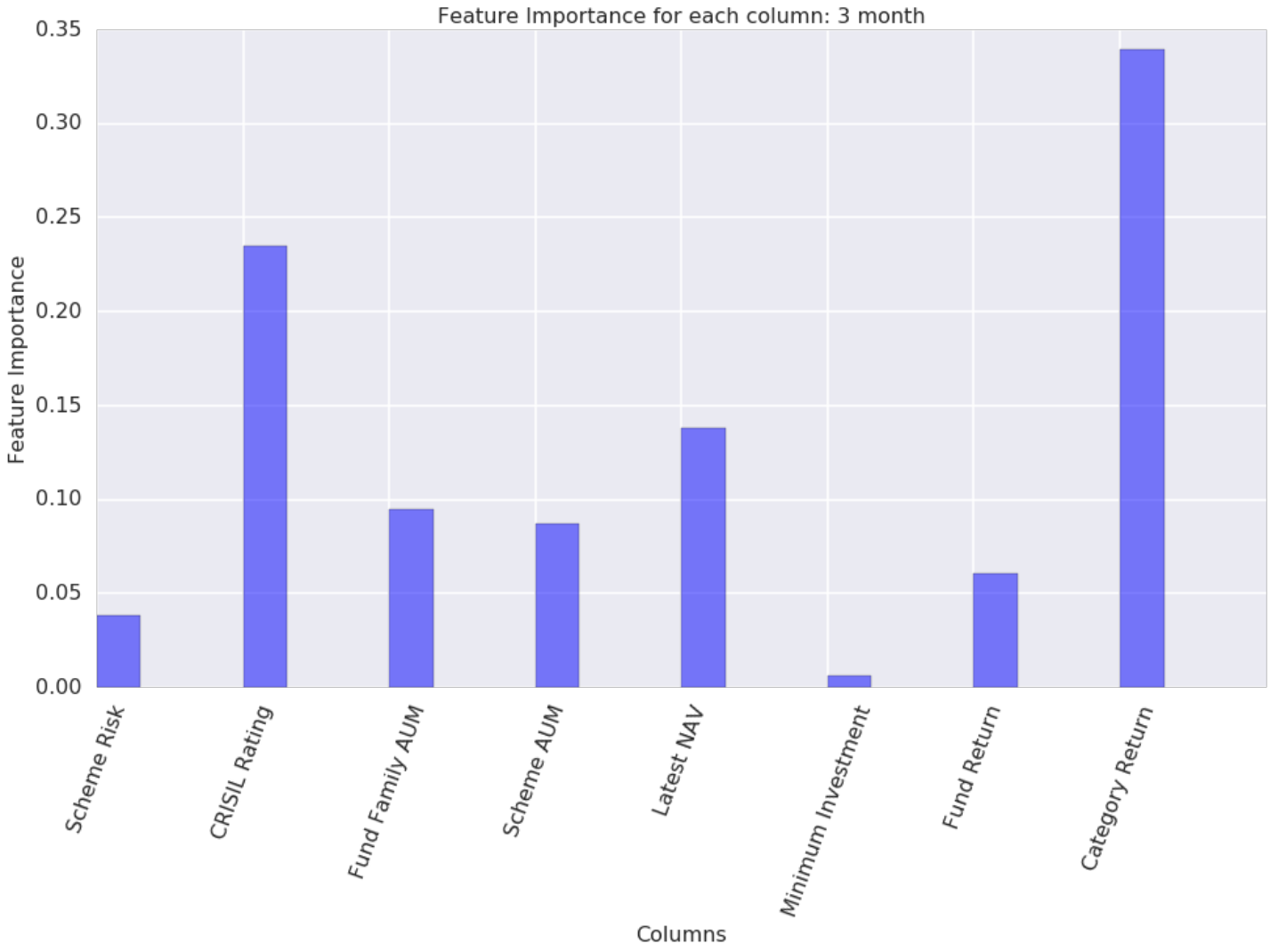
```
# 1 month
plt.figure( figsize = (16, 9) )
plt.bar(index,
        np.delete( feature_imp_1m, [6, 7]),
        bar_width,
        alpha=opacity,
        color='b',
        label='')

plt.xlabel('Columns', fontsize =16)
plt.ylabel('Feature Importance', fontsize =16)
plt.title('Feature Importance for each column: 1 month', fontsize = 16)
plt.xticks(index, np.delete(columns, [6,7]), rotation = 70)
plt.show()
```



In [498]:

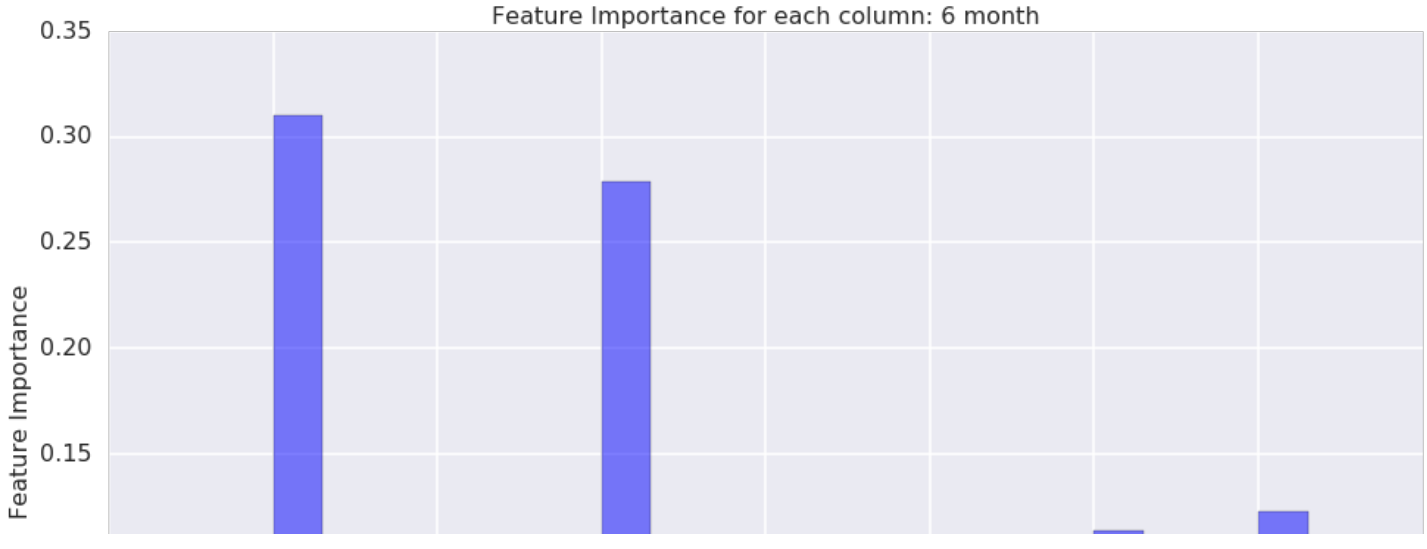
```
# 3 month
plt.figure( figsize = (16, 9) )
plt.bar(index,
        np.delete( feature_imp_3m, [6,7]),
        bar_width,
        alpha=opacity,
        color='b',
        label='')

plt.xlabel('Columns', fontsize =16)
plt.ylabel('Feature Importance', fontsize =16)
plt.title('Feature Importance for each column: 3 month', fontsize = 16)
plt.xticks(index, np.delete(columns, [6,7]), rotation = 70)
```

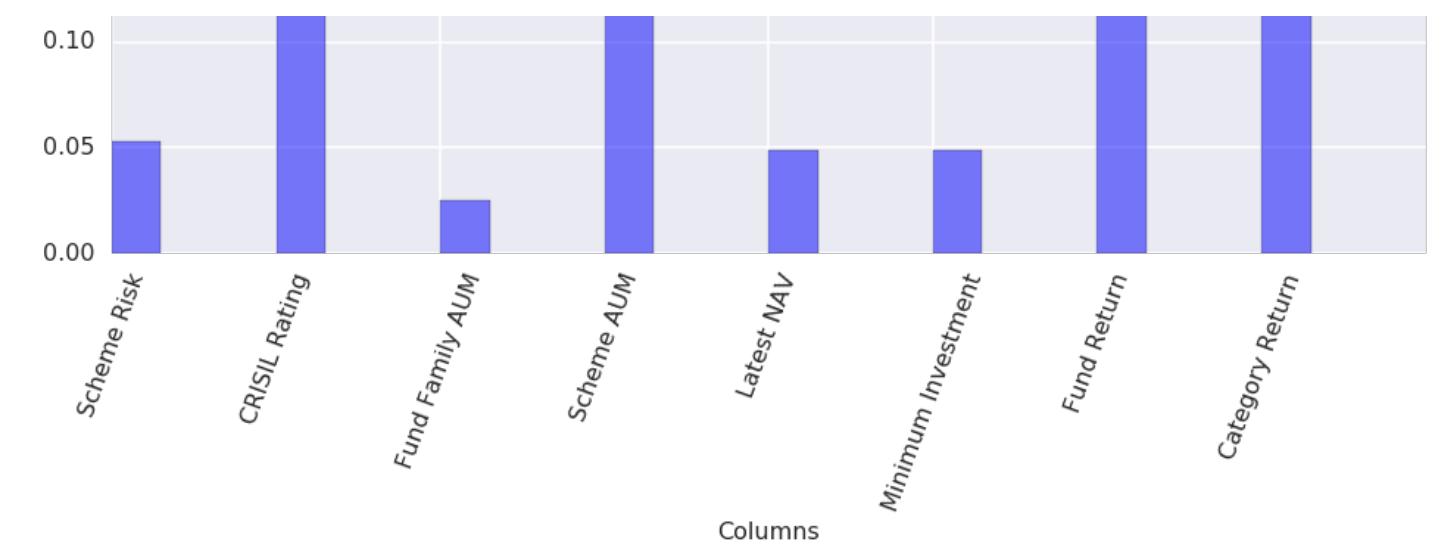
```
plt.show()
```



In [499]:

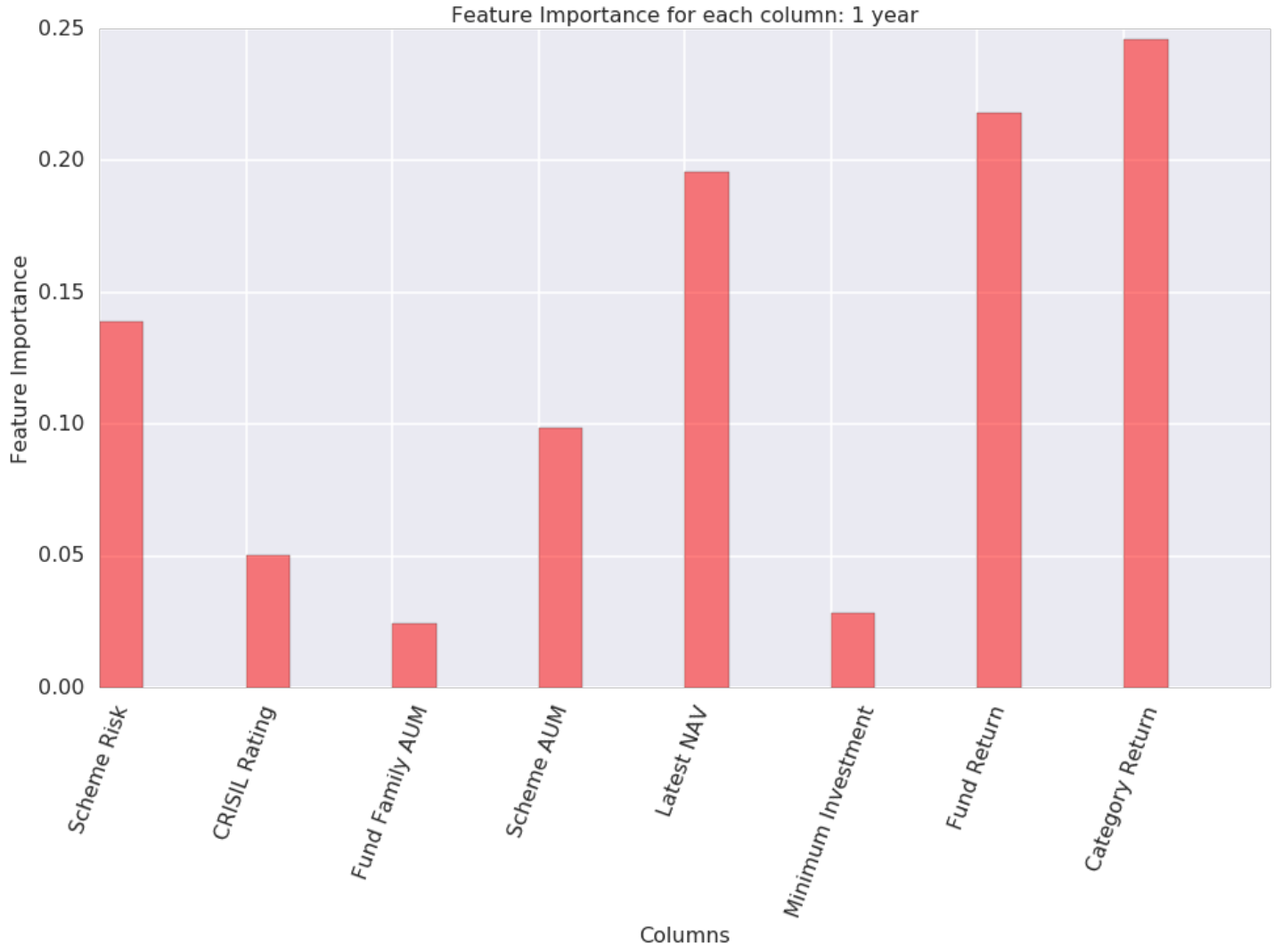
```
# 6 month
plt.figure( figsize = (16, 9) )
plt.bar(index,
        np.delete( feature_imp_6m, [6,7]),
        bar_width,
        alpha=opacity,
        color='b',
        label='')

plt.xlabel('Columns', fontsize =16)
plt.ylabel('Feature Importance', fontsize =16)
plt.title('Feature Importance for each column: 6 month', fontsize = 16)
plt.xticks(index, np.delete(columns, [6,7]), rotation = 70)
plt.show()
```





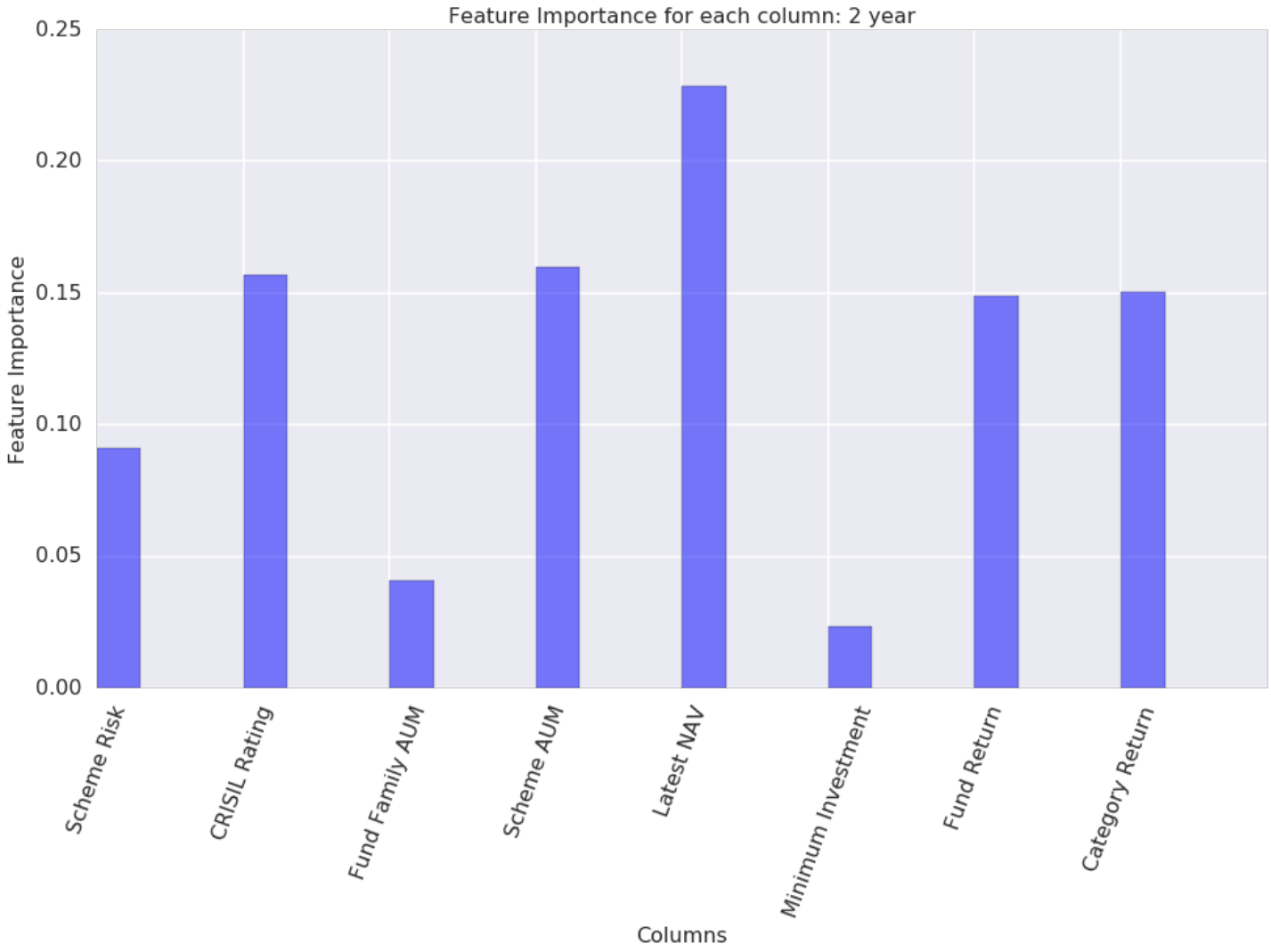
In [500]:

```
# 1 year
plt.figure( figsize = (16, 9) )
plt.bar(index,
        np.delete( feature_imp_1y, [6,7]),
        bar_width,
        alpha=opacity,
        color='r',
        label='')

plt.xlabel('Columns', fontsize =16)
plt.ylabel('Feature Importance', fontsize =16)
plt.title('Feature Importance for each column: 1 year', fontsize = 16)
plt.xticks(index, np.delete(columns, [6,7]), rotation = 70)
plt.show()
```



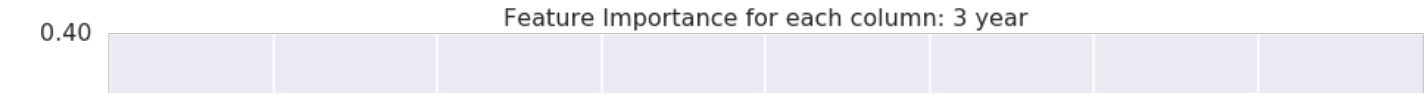
In [501]:

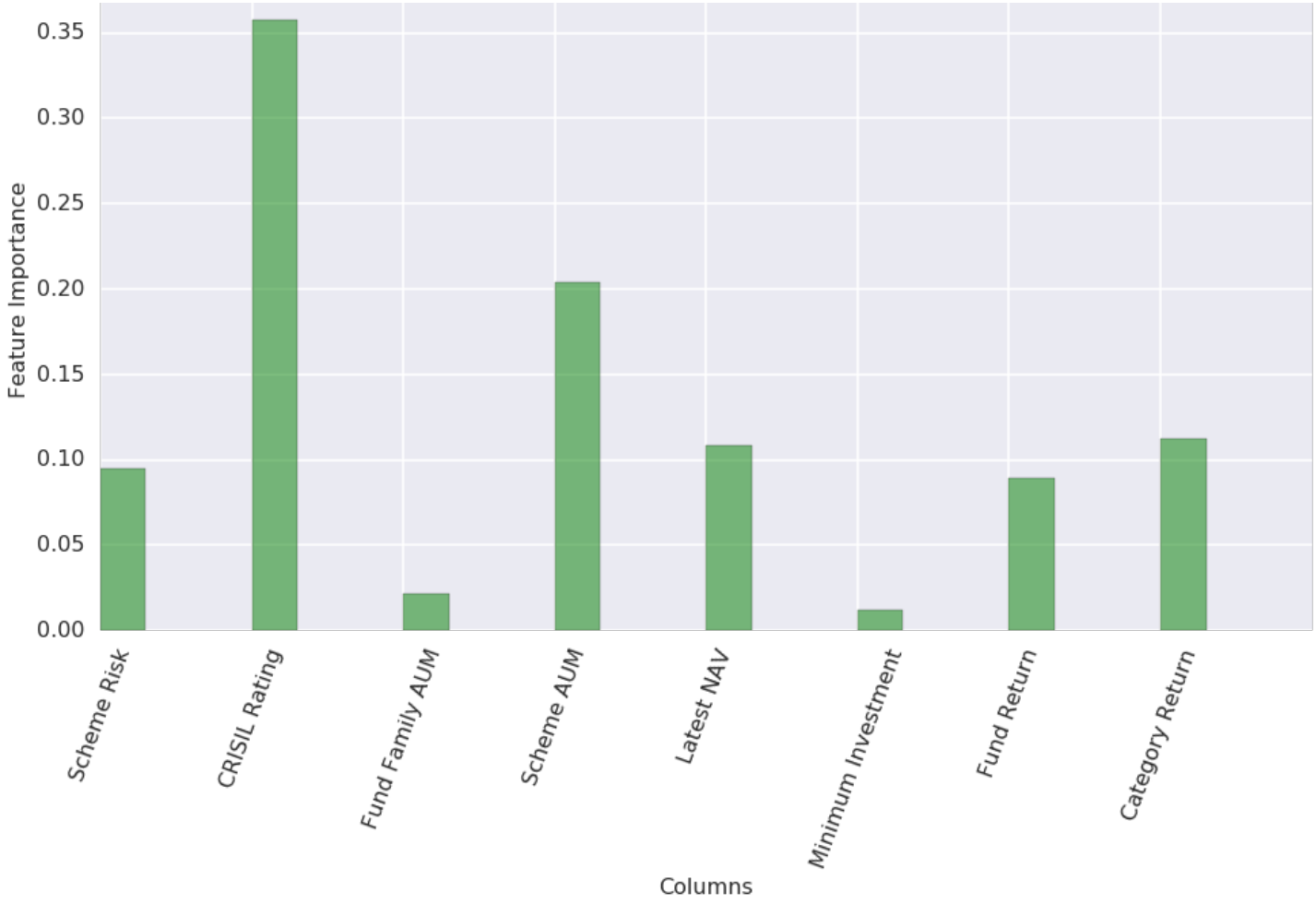
```
# 2 year
plt.figure( figsize = (16, 9) )
plt.bar(index,
        np.delete( feature_imp_2y, [6,7]),
        bar_width,
        alpha=opacity,
        color='b',
        label='')

plt.xlabel('Columns', fontsize =16)
plt.ylabel('Feature Importance', fontsize =16)
plt.title('Feature Importance for each column: 2 year', fontsize = 16)
plt.xticks(index, np.delete(columns, [6,7]), rotation = 70)
plt.show()
```



In [502]:

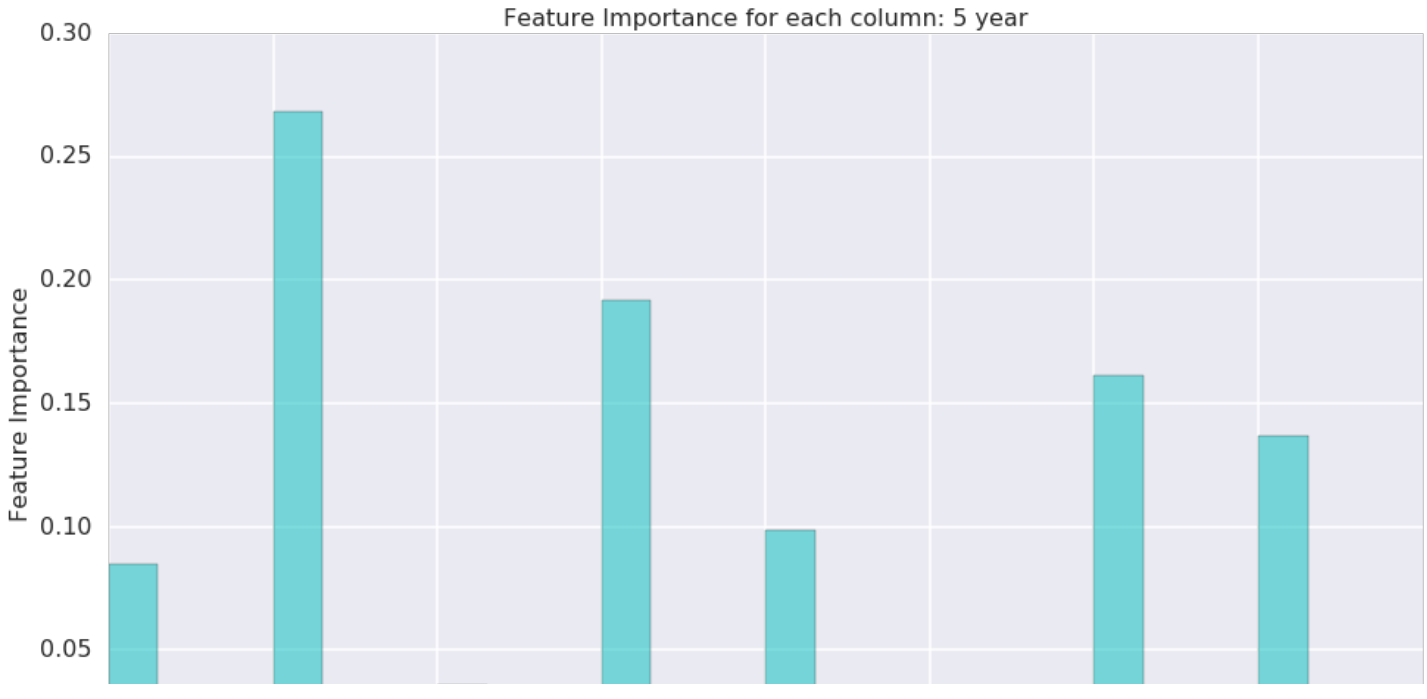
```
# 3 year
plt.figure( figsize = (16, 9) )
plt.bar(index,
        np.delete( feature_imp_3y, [6,7]),
        bar_width,
        alpha=opacity,
        color='g',
        label='')

plt.xlabel('Columns', fontsize =16)
plt.ylabel('Feature Importance', fontsize =16)
plt.title('Feature Importance for each column: 3 year', fontsize = 16)
plt.xticks(index, np.delete(columns, [6,7]), rotation = 70)
plt.show()
```

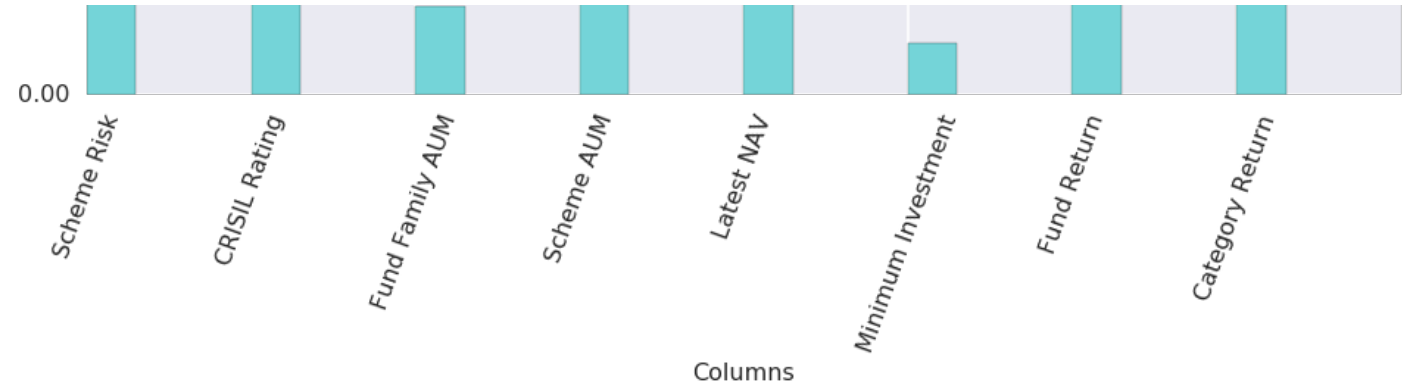




In [503]:

```
# 5 year
plt.figure( figsize = (16, 9) )
plt.bar(index,
        np.delete( feature_imp_5y, [6,7]),
        bar_width,
        alpha=opacity,
        color='c',
        label='')

plt.xlabel('Columns', fontsize =16)
plt.ylabel('Feature Importance', fontsize =16)
plt.title('Feature Importance for each column: 5 year', fontsize = 16)
plt.xticks(index, np.delete(columns, [6,7]), rotation = 70)
plt.show()
```





In [504]:

```
##
# Predict good and bad fund based on Random Forest Classification
##
Y_1m_predicted = clf_1m.predict( X_1m )
Y_3m_predicted = clf_3m.predict( X_3m )
Y_6m_predicted = clf_6m.predict( X_6m )
Y_1y_predicted = clf_1y.predict( X_1y )
Y_2y_predicted = clf_2y.predict( X_2y )
Y_3y_predicted = clf_3y.predict( X_3y )
Y_5y_predicted = clf_5y.predict( X_5y )
```

In [506]:

```
##
# Model Evaluation: Classification Score
##
clf_1m_score = clf_1m.score(X_1m[1000:, :], Y_1m[1000:], sample_weight=None)
clf_3m_score = clf_3m.score(X_3m[1000:, :], Y_3m[1000:], sample_weight=None)
clf_6m_score = clf_6m.score(X_6m[1000:, :], Y_6m[1000:], sample_weight=None)
clf_1y_score = clf_1y.score(X_1y[1000:, :], Y_1y[1000:], sample_weight=None)
clf_2y_score = clf_2y.score(X_2y[1000:, :], Y_2y[1000:], sample_weight=None)
clf_3y_score = clf_3y.score(X_3y[1000:, :], Y_3y[1000:], sample_weight=None)
clf_5y_score = clf_5y.score(X_5y[1000:, :], Y_5y[1000:], sample_weight=None)
```

```
[ 0.  0.  0. ...,  0.  0.  0.]
[ 0.  0.  0. ...,  0.  0.  0.]
[ 0.  0.  0. ...,  0.  0.  0.]
[ 0.  0.  0. ...,  0.  0.  0.]
[ 0.  0.  0. ...,  0.  0.  0.]
[ 1.  1.  1. ...,  0.  0.  0.]
[ 0.  1.  1. ...,  0.  0.  0.]
0.966216216216
0.962837837838
0.942567567568
0.989864864865
0.966216216216
0.942567567568
0.956081081081
```

In [512]:

```
print('Timeframe: {0}\nScore: {1}\n'.format('1m', clf_1m_score) )
print('Timeframe: {0}\nScore: {1}\n'.format('3m', clf_3m_score) )
print('Timeframe: {0}\nScore: {1}\n'.format('6m', clf_6m_score) )
print('Timeframe: {0}\nScore: {1}\n'.format('1y', clf_1y_score) )
print('Timeframe: {0}\nScore: {1}\n'.format('2y', clf_2y_score) )
print('Timeframe: {0}\nScore: {1}\n'.format('3y', clf_3y_score) )
print('Timeframe: {0}\nScore: {1}\n'.format('5y', clf_5y_score) )
```

```
Timeframe: 1m
Score: 0.966216216216

Timeframe: 3m
Score: 0.962837837838

Timeframe: 6m
```

```
Score: 0.942567567568

Timeframe: 1y
Score: 0.989864864865

Timeframe: 2y
Score: 0.966216216216

Timeframe: 3y
Score: 0.942567567568

Timeframe: 5y
Score: 0.956081081081
```

In [526]:

```
##
# List of good funds for each time frame
##
good_funds_1m = [ fund_schemes[k] for (k, v) in enumerate( Y_1m_predicted ) if v == 1.0
]
good_funds_3m = [ fund_schemes[k] for (k, v) in enumerate( Y_3m_predicted ) if v == 1.0
]
good_funds_6m = [ fund_schemes[k] for (k, v) in enumerate( Y_6m_predicted ) if v == 1.0
]
good_funds_1y = [ fund_schemes[k] for (k, v) in enumerate( Y_1y_predicted ) if v == 1.0
]
good_funds_2y = [ fund_schemes[k] for (k, v) in enumerate( Y_2y_predicted ) if v == 1.0
]
good_funds_3y = [ fund_schemes[k] for (k, v) in enumerate( Y_3y_predicted ) if v == 1.0
]
good_funds_5y = [ fund_schemes[k] for (k, v) in enumerate( Y_5y_predicted ) if v == 1.0
]
```

In [542]:

```
good_funds_1m_sort = [ [ fund['scheme_name'], fund[ 'num_fund_ret_1m' ], fund[ 'num_fund
_ret_3m' ], fund[ 'num_fund_ret_6m' ], fund[ 'num_fund_ret_1y' ], fund[ 'num_fund_ret_2y
' ], fund[ 'num_fund_ret_3y' ], fund[ 'num_fund_ret_5y' ], fund[ 'scheme_url' ] ] for fu
nd in good_funds_1m]

good_funds_3m_sort = [ [ fund['scheme_name'], fund[ 'num_fund_ret_1m' ], fund[ 'num_fund
_ret_3m' ], fund[ 'num_fund_ret_6m' ], fund[ 'num_fund_ret_1y' ], fund[ 'num_fund_ret_2y
' ], fund[ 'num_fund_ret_3y' ], fund[ 'num_fund_ret_5y' ], fund[ 'scheme_url' ] ] for fu
nd in good_funds_3m]

good_funds_6m_sort = [ [ fund['scheme_name'], fund[ 'num_fund_ret_1m' ], fund[ 'num_fund
_ret_3m' ], fund[ 'num_fund_ret_6m' ], fund[ 'num_fund_ret_1y' ], fund[ 'num_fund_ret_2y
' ], fund[ 'num_fund_ret_3y' ], fund[ 'num_fund_ret_5y' ], fund[ 'scheme_url' ] ] for fu
nd in good_funds_6m]

good_funds_1y_sort = [ [ fund['scheme_name'], fund[ 'num_fund_ret_1m' ], fund[ 'num_fund
_ret_3m' ], fund[ 'num_fund_ret_6m' ], fund[ 'num_fund_ret_1y' ], fund[ 'num_fund_ret_2y
' ], fund[ 'num_fund_ret_3y' ], fund[ 'num_fund_ret_5y' ], fund[ 'scheme_url' ] ] for fu
nd in good_funds_1y]

good_funds_2y_sort = [ [ fund['scheme_name'], fund[ 'num_fund_ret_1m' ], fund[ 'num_fund
_ret_3m' ], fund[ 'num_fund_ret_6m' ], fund[ 'num_fund_ret_1y' ], fund[ 'num_fund_ret_2y
' ], fund[ 'num_fund_ret_3y' ], fund[ 'num_fund_ret_5y' ], fund[ 'scheme_url' ] ] for fu
nd in good_funds_2y]

good_funds_3y_sort = [ [ fund['scheme_name'], fund[ 'num_fund_ret_1m' ], fund[ 'num_fund
_ret_3m' ], fund[ 'num_fund_ret_6m' ], fund[ 'num_fund_ret_1y' ], fund[ 'num_fund_ret_2y
' ], fund[ 'num_fund_ret_3y' ], fund[ 'num_fund_ret_5y' ], fund[ 'scheme_url' ] ] for fu
nd in good_funds_3y]

good_funds_5y_sort = [ [ fund['scheme_name'], fund[ 'num_fund_ret_1m' ], fund[ 'num_fund
_ret_3m' ], fund[ 'num_fund_ret_6m' ], fund[ 'num_fund_ret_1y' ], fund[ 'num_fund_ret_2y
' ], fund[ 'num_fund_ret_3y' ], fund[ 'num_fund_ret_5y' ], fund[ 'scheme_url' ] ] for fu
nd in good_funds_5y]
```

In [544]:

```
good_funds_1m_sort.sort(key = lambda x: x[1], reverse=True)
    good_funds_3m_sort.sort(key = lambda x: x[2], reverse=True)
    good_funds_6m_sort.sort(key = lambda x: x[3], reverse=True)
    good_funds_1y_sort.sort(key = lambda x: x[4], reverse=True)
    good_funds_2y_sort.sort(key = lambda x: x[5], reverse=True)
    good_funds_3y_sort.sort(key = lambda x: x[6], reverse=True)
    good_funds_5y_sort.sort(key = lambda x: x[7], reverse=True)

    print( '## Top 5 funds for a timeframe of 1 month:')
    print( good_funds_1m_sort[:5] )

    print( '## Top 5 funds for a timeframe of 3 months:')
    print( good_funds_3m_sort[:5] )

    print( '## Top 5 funds for a timeframe of 6 months:')
    print( good_funds_6m_sort[:5] )

    print( '## Top 5 funds for a timeframe of 1 year:')
    print( good_funds_1y_sort[:5] )

    print( '## Top 5 funds for a timeframe of 2 years:')
    print( good_funds_2y_sort[:5] )

    print( '## Top 5 funds for a timeframe of 3 years:')
    print( good_funds_3y_sort[:5] )

    print( '## Top 5 funds for a timeframe of 5 years:')
    print( good_funds_5y_sort[:5] )
```

```
## Top 5 funds for a timeframe of 1 month:
[[u'UTI Infrastructure Fund (G)', 8.5, -6.0, -11.2, -16.8, 11.3, 11.6, 2.2, 'http://www.m
oneycontrol.com/mutual-funds/nav/uti-infrastructure-fund/MUT065'], [u'HDFC Tax Saver (G)'
, 7.9, -4.5, -7.2, -12.5, 12.3, 15.6, 8.5, 'http://www.moneycontrol.com/mutual-funds/nav/
hdfc-tax-saver/MZU017'], [u'UTI Dividend Yield Fund (G)', 7.0, -1.9, -4.7, -8.8, 10.3, 11
.3, 6.2, 'http://www.moneycontrol.com/mutual-funds/nav/uti-dividend-yield-fund/MUT070'],
[u'IDFC Sterling Equity Fund - RP (G)', 5.4, -8.7, -7.2, -13.9, 18.2, 17.2, 12.0, 'http:/
/www.moneycontrol.com/mutual-funds/nav/idfc-sterling-equity-fund-regular-plan/MAG162'], [
u'UTI Opportunities Fund (G)', 5.3, -1.9, -7.0, -12.9, 9.7, 12.8, 9.5, 'http://www.moneyc
ontrol.com/mutual-funds/nav/uti-opportunities-fund/MUT072']]
## Top 5 funds for a timeframe of 3 months:
[[u'ICICI Pru Long Term Gilt (G)', 3.7, 3.3, 2.5, 5.7, 12.8, 8.2, 8.4, 'http://www.moneyc
ontrol.com/mutual-funds/nav/icici-prudential-long-term-gilt-fund/MPI008'], [u'Birla Sun L
ife GSec - LTF (G)', 3.8, 3.3, 2.5, 5.6, 12.4, 8.6, 9.2, 'http://www.moneycontrol.com/mut
ual-funds/nav/birla-sun-life-g-sec-fund-long-term/MAC017'], [u'Kotak Gilt Invt - Regular
(G)', 3.2, 3.1, 2.5, 5.4, 12.1, 7.6, 8.9, 'http://www.moneycontrol.com/mutual-funds/nav/k
otak-gilt-investment-plan-regular/MKM002'], [u'Franklin (I) Govt Sec -CP (G)', 3.1, 3.0,
2.2, 5.3, 13.2, 8.7, 8.2, 'http://www.moneycontrol.com/mutual-funds/nav/franklin-india-go
vt-sec-composite-plan/MTE005'], [u'HDFC High Interest - STP (G)', 1.6, 2.3, 3.5, 7.2, 9.2
, 8.6, 8.9, 'http://www.moneycontrol.com/mutual-funds/nav/hdfc-high-interest-fund-short-t
erm-plan/MZU026']]
## Top 5 funds for a timeframe of 6 months:
[[u'Franklin Ultra SBF - SIP (G)', 1.0, 2.3, 4.6, 9.4, 9.7, 9.9, 9.9, 'http://www.moneyco
ntrol.com/mutual-funds/nav/franklin-india-ultra-short-bond-fund-super-institutional-plan/
MTE188'], [u'UTI Treasury Advtg -Inst (G)', 1.0, 2.2, 4.2, 8.6, 9.0, 9.2, 9.3, 'http://ww
w.moneycontrol.com/mutual-funds/nav/uti-treasury-advantage-fund-institutional-plan/MUT119
'], [u'HDFC Floating Rate Inc.-STP-WP (G)', 1.0, 2.1, 4.1, 8.5, 9.0, 9.1, 9.2, 'http://ww
w.moneycontrol.com/mutual-funds/nav/hdfc-floating-rate-income-fund-stp-wholesale-plan/MHD
247'], [u'ICICI Pru Flexi Income (G)', 1.1, 2.2, 4.1, 8.6, 9.0, 9.3, 9.3, 'http://www.mon
eycontrol.com/mutual-funds/nav/icici-prudential-flexible-income-plan/MPI036'], [u'Birla S
L Saving Fund - IP (G)', 1.0, 2.2, 4.1, 8.7, 9.1, 9.3, 9.4, 'http://www.moneycontrol.com/
mutual-funds/nav/birla-sun-life-savings-fund-institutional-plan-c-/MBS037']]
## Top 5 funds for a timeframe of 1 year:
[[u'SBI Savings Fund - Direct (G)', 1.1, 2.3, 4.2, 8.8, 9.2, 9.4, None, 'http://www.money
control.com/mutual-funds/nav/sbi-savings-fund-direct-plan/MSB555'], [u'Franklin (I) Savin
gs Plus - IP (G)', 0.6, 2.0, 3.9, 8.7, 9.2, 9.0, 9.0, 'http://www.moneycontrol.com/mutual
-funds/nav/franklin-india-savings-plus-fund-institutional-plan/MTE088'], [u'Franklin (I)
Savings Plus - DP (G)', 1.0, 2.2, 4.2, 8.6, 9.2, 9.3, None, 'http://www.moneycontrol.com/
mutual-funds/nav/franklin-india-savings-plus-fund-direct-plan/MTE365'], [u'Reliance Liqui
d - Cash -Direct (G)', 0.8, 2.1, 4.1, 8.4, 8.8, 8.9, None, 'http://www.moneycontrol.com/m
utual-funds/nav/reliance-liquid-fund-cash-plan-direct-plan/MRC972'], [u'Reliance Medium T
```

```
erm Fund (G)', 1.0, 2.1, 3.9, 8.4, 8.9, 8.8, 9.0, 'http://www.moneycontrol.com/mutual-fun
ds/nav/reliance-medium-term-fund/MRC010']]
## Top 5 funds for a timeframe of 2 years:
[[u'Birla Sun Life Midcap Fund (G)', 6.5, -6.4, -6.4, -4.9, 26.7, 23.8, 13.9, 'http://www
.moneycontrol.com/mutual-funds/nav/birla-sun-life-midcap-fund/MBS027'], [u'DSP-BR Small &
Mid Cap -RP (G)', 8.2, -5.8, -2.1, -4.1, 26.3, 25.6, 14.7, 'http://www.moneycontrol.com/m
utual-funds/nav/dsp-blackrock-small-and-mid-cap-fund/MDS056'], [u'Reliance Mid & Small Ca
p Fund (G)', 4.6, -12.0, -8.7, -9.7, 26.1, 26.0, 14.5, 'http://www.moneycontrol.com/mutua
l-funds/nav/reliance-mid-small-cap-fund/MRC110'], [u'SBI Emerging Busi (G)', 7.4, -4.2, 0
.1, -2.5, 23.3, 18.6, 17.0, 'http://www.moneycontrol.com/mutual-funds/nav/sbi-emerging-bu
sinesses-fund/MSB059'], [u'ICICI Pru Gilt Inv Plan - PF (G)', 4.0, 3.6, 2.8, 5.9, 13.5, 9
.1, 8.5, 'http://www.moneycontrol.com/mutual-funds/nav/icici-prudential-gilt-investment-p
lan-pf-option/MPI067']]
## Top 5 funds for a timeframe of 3 years:
[[u'Birla SL Pure Value Fund (G)', 7.2, -4.9, -1.8, -2.0, 29.1, 30.0, 17.6, 'http://www.m
oneycontrol.com/mutual-funds/nav/birla-sun-life-pure-value-fund/MBS267'], [u'Reliance Mid
& Small Cap Fund (G)', 4.6, -12.0, -8.7, -9.7, 26.1, 26.0, 14.5, 'http://www.moneycontrol
.com/mutual-funds/nav/reliance-mid-small-cap-fund/MRC110'], [u'DSP-BR Small & Mid Cap -RP
(G)', 8.2, -5.8, -2.1, -4.1, 26.3, 25.6, 14.7, 'http://www.moneycontrol.com/mutual-funds/
nav/dsp-blackrock-small-and-mid-cap-fund/MDS056'], [u'Birla Sun Life Midcap Fund (G)', 6.
5, -6.4, -6.4, -4.9, 26.7, 23.8, 13.9, 'http://www.moneycontrol.com/mutual-funds/nav/birl
a-sun-life-midcap-fund/MBS027'], [u'IDFC Premier Equity - Regular (G)', 4.7, -5.9, -5.2,
-9.0, 20.8, 22.5, 16.3, 'http://www.moneycontrol.com/mutual-funds/nav/idfc-premier-equity
-fund-regular-plan/MAG094']]
## Top 5 funds for a timeframe of 5 years:
[[u'Birla SL Pure Value Fund (G)', 7.2, -4.9, -1.8, -2.0, 29.1, 30.0, 17.6, 'http://www.m
oneycontrol.com/mutual-funds/nav/birla-sun-life-pure-value-fund/MBS267'], [u'SBI Emerging
Busi (G)', 7.4, -4.2, 0.1, -2.5, 23.3, 18.6, 17.0, 'http://www.moneycontrol.com/mutual-fu
nds/nav/sbi-emerging-businesses-fund/MSB059'], [u'Reliance Mid & Small Cap Fund (G)', 4.6
, -12.0, -8.7, -9.7, 26.1, 26.0, 14.5, 'http://www.moneycontrol.com/mutual-funds/nav/reli
ance-mid-small-cap-fund/MRC110'], [u'Birla Sun Life Midcap Fund (G)', 6.5, -6.4, -6.4, -4
.9, 26.7, 23.8, 13.9, 'http://www.moneycontrol.com/mutual-funds/nav/birla-sun-life-midcap
-fund/MBS027'], [u'ICICI Pru Balanced Adv (G)', 4.1, -1.6, -2.1, 0.2, 12.9, 15.2, 12.8, '
http://www.moneycontrol.com/mutual-funds/nav/icici-prudential-balanced-advantage-fund/MPI
126']]
```

In [550]:

```
# Demonstrate class imbalance

# Print the number of good samples
print( sum(Y_1m[1000:]), sum(Y_3m), sum(Y_6m), sum(Y_1y), sum(Y_2y), sum(Y_3y), sum(Y_5y)
)

# Print the number of total samples
print( len(Y_1m[1000:]), len(Y_3m), len(Y_6m), len(Y_1y), len(Y_2y), len(Y_3y), len(Y_5y)
)
```

```
(12.0, 52.0, 61.0, 72.0, 120.0, 139.0, 107.0)
(296, 1296, 1296, 1296, 1296, 1296, 1296)
```

In [ ]:

```
```