1. Load the dataset

Libraries we use

1. pandas
2. numpy
3. matplotlib
4. sklearn
5. scipy
6. seaborn
7. mpl_toolkits.

In [1]:

```
import pandas as pd
# Code to read csv file into Colaboratory:
# !pip install -U -q PyDrive
# from pydrive.auth import GoogleAuth
# from pydrive.drive import GoogleDrive
# from google.colab import auth
# from oauth2client.client import GoogleCredentials
# # Authenticate and create the PyDrive client.
# auth.authenticate_user()
# gauth = GoogleAuth()
# gauth.credentials = GoogleCredentials.get_application_default()
# drive = GoogleDrive(gauth)

# link = 'https://drive.google.com/open?id=1kR3TcMccX8m3aScfno4wjY15vkqH1s_a'
# fluff, id = link.split('=')
# print (id) # Verify that you have everything after '='
# downloaded = drive.CreateFile({'id':id})
# downloaded.GetContentFile('test_file1.txt')

# #https://www.kaggle.com/theoviel/load-the-totality-of-the-data
dtypes = {
        'MachineIdentifier':                                    'category',
        'ProductName':                                          'category',
        'EngineVersion':                                        'category',
        'AppVersion':                                           'category',
        'AvSigVersion':                                         'category',
        'IsBeta':                                               'int8',
        'RtpStateBitfield':                                     'float16',
        'IsSxsPassiveMode':                                     'int8',
        'DefaultBrowsersIdentifier':                            'float32',
        'AVProductStatesIdentifier':                            'float32',
        'AVProductsInstalled':                                  'float16',
        'AVProductsEnabled':                                    'float16',
        'HasTpm':                                               'int8',
        'CountryIdentifier':                                    'int16',
        'CityIdentifier':                                       'float32',
        'OrganizationIdentifier':                               'float16',
        'GeoNameIdentifier':                                    'float16',
        'LocaleEnglishNameIdentifier':                          'int16',
        'Platform':                                             'category',
        'Processor':                                            'category',
        'OsVer':                                                'category',
        'OsBuild':                                              'int16',
        'OsSuite':                                              'int16',
        'OsPlatformSubRelease':                                 'category',
        'OsBuildLab':                                           'category',
        'SkuEdition':                                           'category',
        'IsProtected':                                          'float16',
        'AutoSampleOptIn':                                      'int8',
        'PuaMode':                                              'category',
        'SMode':                                                'float16',
        'IeVerIdentifier':                                      'float16',
        'SmartScreen':                                          'category',
        'Firewall':                                             'float16',
        'UacLuaenable':                                         'float64', # was 'float32'
        'Census_MDC2FormFactor':                                'category',
        'Census_DeviceFamily':                                  'category',
        'Census_OEMNameIdentifier':                             'float32', # was 'float16'
        'Census_OEMModelIdentifier':                            'float32',
        'Census_ProcessorCoreCount':                            'float16',
        'Census_ProcessorManufacturerIdentifier':               'float16',
        'Census_ProcessorModelIdentifier':                      'float32', # was 'float16'
        'Census_ProcessorClass':                                'category',
        'Census_PrimaryDiskTotalCapacity':                      'float64', # was 'float32'
        'Census_PrimaryDiskTypeName':                           'category',
        'Census_SystemVolumeTotalCapacity':                     'float64', # was 'float32'
        'Census_HasOpticalDiskDrive':                           'int8',
        'Census_TotalPhysicalRAM':                              'float32',
        'Census_ChassisTypeName':                               'category',
        'Census_InternalPrimaryDiagonalDisplaySizeInInches':    'float32', # was 'float16'
        'Census_InternalPrimaryDisplayResolutionHorizontal':    'float32', # was 'float16'
        'Census_InternalPrimaryDisplayResolutionVertical':      'float32', # was 'float16'
        'Census_PowerPlatformRoleName':                         'category',
        'Census_InternalBatteryType':                           'category',
        'Census_InternalBatteryNumberOfCharges':                'float64', # was 'float32'
        'Census_OSVersion':                                     'category',
        'Census_OSArchitecture':                                'category',
        'Census_OSBranch':                                      'category',
        'Census_OSBuildNumber':                                 'int16',
        'Census_OSBuildRevision':                               'int32',
        'Census_OSEdition':                                     'category',
        'Census_OSSkuName':                                     'category',
        'Census_OSInstallTypeName':                             'category',
        'Census_OSInstallLanguageIdentifier':                   'float16',
        'Census_OSUILocaleIdentifier':                          'int16',
        'Census_OSWUAutoUpdateOptionsName':                     'category',
        'Census_IsPortableOperatingSystem':                     'int8',
        'Census_GenuineStateName':                              'category',
        'Census_ActivationChannel':                             'category',
        'Census_IsFlightingInternal':                           'float16',
        'Census_IsFlightsDisabled':                             'float16',
        'Census_FlightRing':                                    'category',
        'Census_ThresholdOptIn':                                'float16',
         'Census_FirmwareManufacturerIdentifier':                'float16',
        'Census_FirmwareVersionIdentifier':                     'float32',
        'Census_IsSecureBootEnabled':                           'int8',
        'Census_IsWIMBootEnabled':                              'float16',
        'Census_IsVirtualDevice':                               'float16',
        'Census_IsTouchEnabled':                                'int8',
        'Census_IsPenCapable':                                  'int8',
        'Census_IsAlwaysOnAlwaysConnectedCapable':              'float16',
        'Wdft_IsGamer':                                         'float16',
        'Wdft_RegionIdentifier':                                'float16',
        'HasDetections':                                        'int8'
        }
train = pd.read_csv('test_file1.txt', delimiter=',', dtype=dtypes)
```

In [2]:

```
train.shape
```

Out[2]:

```
(499999, 83)
```

In [3]:

```
numerics = ['int8', 'int16', 'int32', 'int64', 'float16', 'float32', 'float64']
num_columns = [c for c,v in dtypes.items() if v in numerics]
cat_columns = [c for c,v in dtypes.items() if v not in numerics]

stats = []
for col in train.columns:
    stats.append((col, train[col].nunique(), train[col].isnull().sum() * 100 / train.shape[0], trai
n[col].value_counts(normalize=True, dropna=False).values[0] * 100, train[col].dtype))

stats_df = pd.DataFrame(stats, columns=['Feature', 'Unique_values', 'missing_values(%)', 'skewness'
, 'type'])
stats_df.sort_values('missing_values(%)', ascending=False)
```

Out[3]:

	Feature	Unique_values	missing_values(%)	skewness	type
28	PuaMode	1	99.975600	99.975600	category
41	Census_ProcessorClass	3	99.579999	99.579999	category
8	DefaultBrowsersIdentifier	557	95.139590	95.139590	float32
68	Census_IsFlightingInternal	2	83.030966	83.030966	float16
52	Census_InternalBatteryType	28	71.028342	71.028342	category
71	Census_ThresholdOptIn	2	63.502727	63.502727	float16
75	Census_IsWIMBootEnabled	1	63.414727	63.414727	float16
31	SmartScreen	12	35.659071	48.334297	category
15	OrganizationIdentifier	43	30.871662	47.089494	float16
29	SMode	2	6.011612	93.945388	float16
14	CityIdentifier	37307	3.641807	3.641807	float32
80	Wdft_IsGamer	2	3.418207	69.285539	float16
81	Wdft_RegionIdentifier	15	3.418207	20.205240	float16
53	Census_InternalBatteryNumberOfCharges	5188	3.025006	56.580713	float64
72	Census_FirmwareManufacturerIdentifier	304	2.052004	30.239660	float16
73	Census_FirmwareVersionIdentifier	23544	1.791204	1.791204	float32
69	Census_IsFlightsDisabled	2	1.779404	98.219796	float16
37	Census_OEMModelIdentifier	40892	1.131602	3.418607	float32
36	Census_OEMNameIdentifier	1620	1.054002	14.490429	float32
32	Firewall	2	1.035802	96.835394	float16
46	Census_TotalPhysicalRAM	561	0.905802	45.957492	float32
79	Census_IsAlwaysOnAlwaysConnectedCapable	2	0.794802	93.551387	float16
30	IeVerIdentifier	188	0.675001	43.514287	float16
62	Census_OSInstallLanguageIdentifier	39	0.663601	35.668271	float16
42	Census_PrimaryDiskTotalCapacity	1133	0.593201	31.881264	float64
44	Census_SystemVolumeTotalCapacity	142066	0.593201	0.593201	float64
48	Census_InternalPrimaryDiagonalDisplaySizeInInches	507	0.548001	34.123868	float32
49	Census_InternalPrimaryDisplayResolutionHorizontal	509	0.546801	50.628101	float32
50	Census_InternalPrimaryDisplayResolutionVertical	542	0.546801	55.734511	float32
40	Census_ProcessorModelIdentifier	2266	0.470001	3.251007	float32
...	...	...	...	...	...
77	Census_IsTouchEnabled	2	0.000000	87.422375	int8
70	Census_FlightRing	7	0.000000	93.687187	category
74	Census_IsSecureBootEnabled	2	0.000000	51.323103	int8
59	Census_OSEdition	21	0.000000	38.992678	category
0	MachineIdentifier	499999	0.000000	0.000200	category
57	Census_OSBuildNumber	65	0.000000	44.892090	int16
20	OsVer	18	0.000000	96.743993	category
2	EngineVersion	55	0.000000	43.135086	category
3	AppVersion	93	0.000000	57.725115	category
4	AvSigVersion	6506	0.000000	1.161402	category
5	IsBeta	2	0.000000	99.999000	int8
7	IsSxsPassiveMode	2	0.000000	98.271397	int8
12	HasTpm	2	0.000000	98.782198	int8
13	CountryIdentifier	222	0.000000	4.459409	int16
17	LocaleEnglishNameIdentifier	233	0.000000	23.474447	int16
18	Platform	4	0.000000	96.588593	category
19	Processor	3	0.000000	90.902182	category

	Feature	Unique_values	missing_values(%)	skewness	type
21	OsBuild	51	0.000000	43.855288	int16
56	Census_OSBranch	16	0.000000	44.895090	category
22	OsSuite	10	0.000000	62.395925	int16
23	OsPlatformSubRelease	9	0.000000	43.855488	category
25	SkuEdition	8	0.000000	61.873524	category
27	AutoSampleOptIn	2	0.000000	99.997200	int8
34	Census_MDC2FormFactor	12	0.000000	64.153328	category
35	Census_DeviceFamily	2	0.000000	99.838400	category
1	ProductName	5	0.000000	98.927598	category
45	Census_HasOpticalDiskDrive	2	0.000000	92.322385	int8
54	Census_OSVersion	307	0.000000	15.798632	category
55	Census_OSArchitecture	3	0.000000	90.903582	category
82	HasDetections	2	0.000000	50.076700	int8

83 rows × 5 columns

In [4]:

```
train[0:3]
```

Out[4]:

	MachineIdentifier	ProductName	EngineVersion	AppVersion	AvSigVersion	IsBeta	RtpStateBitfield	IsSxsPass
0	0000028988387b115f69f31a3bf04f09	win8defender	1.1.15100.1	4.18.1807.18075	1.273.1735.0	0	7.0	
1	000007535c3f730efa9ea0b7ef1bd645	win8defender	1.1.14600.4	4.13.17134.1	1.263.48.0	0	7.0	
2	000007905a28d863f6d0d597892cd692	win8defender	1.1.15100.1	4.18.1807.18075	1.273.1341.0	0	7.0	

3 rows × 83 columns

2. Preprocessing

As we cited in the report, in preprocessing, we used an external code from “Load the Totality of the Data.” Kaggle, © 2019 Kaggle Inc., www.kaggle.com/theoviel/load-the-totality-of-the-data.

In [5]:

```
drop_features = list()
```

a) Select mostly missing features which have more than 95% of missing values

In [6]:

```
missing = (train.isnull().sum()/train.shape[0]).sort_values(ascending=False)
print(missing)
```

```
PuaMode                                              0.999756
Census_ProcessorClass                                0.995800
DefaultBrowsersIdentifier                            0.951396
Census_IsFlightingInternal                           0.830310
Census_InternalBatteryType                           0.710283
Census_ThresholdOptIn                                0.635027
Census_IsWIMBootEnabled                              0.634147
SmartScreen                                          0.356591
OrganizationIdentifier                               0.308717
SMode                                                0.060116
CityIdentifier                                       0.036418
Wdft_IsGamer                                         0.034182
Wdft_RegionIdentifier                                0.034182
Census_InternalBatteryNumberOfCharges                0.030250
Census_FirmwareManufacturerIdentifier                0.020520
```

```
Census_FirmwareVersionIdentifier                     0.017912
Census_IsFlightsDisabled                             0.017794
Census_OEMModelIdentifier                            0.011316
Census_OEMNameIdentifier                             0.010540
Firewall                                             0.010358
Census_TotalPhysicalRAM                              0.009058
Census_IsAlwaysOnAlwaysConnectedCapable              0.007948
IeVerIdentifier                                      0.006750
Census_OSInstallLanguageIdentifier                   0.006636
Census_PrimaryDiskTotalCapacity                      0.005932
Census_SystemVolumeTotalCapacity                     0.005932
Census_InternalPrimaryDiagonalDisplaySizeInInches    0.005480
Census_InternalPrimaryDisplayResolutionHorizontal    0.005468
Census_InternalPrimaryDisplayResolutionVertical      0.005468
Census_ProcessorModelIdentifier                      0.004700
                                                       ...
ProductName                                          0.000000
HasTpm                                               0.000000
OsBuild                                              0.000000
IsBeta                                               0.000000
OsSuite                                              0.000000
IsSxsPassiveMode                                     0.000000
HasDetections                                        0.000000
SkuEdition                                           0.000000
Census_OSInstallTypeName                             0.000000
Census_IsPenCapable                                  0.000000
Census_IsTouchEnabled                                0.000000
Census_IsSecureBootEnabled                           0.000000
Census_FlightRing                                    0.000000
Census_ActivationChannel                             0.000000
Census_GenuineStateName                              0.000000
Census_IsPortableOperatingSystem                     0.000000
Census_OSWUAutoUpdateOptionsName                     0.000000
Census_OSUILocaleIdentifier                          0.000000
Census_OSSkuName                                     0.000000
AutoSampleOptIn                                      0.000000
Census_OSEdition                                     0.000000
Census_OSBuildRevision                               0.000000
Census_OSBuildNumber                                 0.000000
Census_OSBranch                                      0.000000
Census_OSArchitecture                                0.000000
Census_OSVersion                                     0.000000
Census_HasOpticalDiskDrive                           0.000000
Census_DeviceFamily                                  0.000000
Census_MDC2FormFactor                                0.000000
MachineIdentifier                                    0.000000
Length: 83, dtype: float64
```

There are 2 columns which have more than 99% of missing values.

In [7]:

```
drop_features.append('PuaMode')
drop_features.append('Census_ProcessorClass')
```

b) Select too skewed columns

In [8]:

```
skew_data = pd.DataFrame([{'columns': c, 'unique': train[c].nunique(),
                           'skewness': train[c].value_counts(normalize=True).values[0]
                          } for c in train.columns])
skew_data = skew_data.sort_values('skewness', ascending=False)
skew_data
```

Out[8]:

	columns	skewness	unique
28	PuaMode	1.000000	1
75	Census_IsWIMBootEnabled	1.000000	1

	columns	skewness	unique
69	Census_IsFlightsDisabled	0.999992	2
5	IsBeta	0.999990	2
68	Census_IsFlightingInternal	0.999988	2
27	AutoSampleOptIn	0.999972	2
71	Census_ThresholdOptIn	0.999710	2
29	SMode	0.999542	2
65	Census_IsPortableOperatingSystem	0.999364	2
35	Census_DeviceFamily	0.998384	2
33	UacLuaenable	0.994045	5
76	Census_IsVirtualDevice	0.993037	2
1	ProductName	0.989276	5
12	HasTpm	0.987822	2
7	IsSxsPassiveMode	0.982714	2
32	Firewall	0.978489	2
11	AVProductsEnabled	0.974016	5
6	RtpStateBitfield	0.973286	6
20	OsVer	0.967440	18
18	Platform	0.965886	4
78	Census_IsPenCapable	0.962024	2
26	IsProtected	0.945313	2
79	Census_IsAlwaysOnAlwaysConnectedCapable	0.943009	2
70	Census_FlightRing	0.936872	7
45	Census_HasOpticalDiskDrive	0.923224	2
55	Census_OSArchitecture	0.909036	3
19	Processor	0.909022	3
66	Census_GenuineStateName	0.883184	4
39	Census_ProcessorManufacturerIdentifier	0.882139	4
77	Census_IsTouchEnabled	0.874224	2
...	...	...	...
57	Census_OSBuildNumber	0.448921	65
64	Census_OSWUAutoUpdateOptionsName	0.442477	6
23	OsPlatformSubRelease	0.438555	9
21	OsBuild	0.438553	51
30	IeVerIdentifier	0.438100	188
2	EngineVersion	0.431351	55
24	OsBuildLab	0.409786	464
59	Census_OSEdition	0.389927	21
60	Census_OSSkuName	0.389921	20
62	Census_OSInstallLanguageIdentifier	0.359065	39
63	Census_OSUILocaleIdentifier	0.355709	95
48	Census_InternalPrimaryDiagonalDisplaySizeInInches	0.343119	507
42	Census_PrimaryDiskTotalCapacity	0.320715	1133
72	Census_FirmwareManufacturerIdentifier	0.308732	304
61	Census_OSInstallTypeName	0.292515	9
17	LocaleEnglishNameIdentifier	0.234744	233
81	Wdft_RegionIdentifier	0.209203	15
16	GeoNameIdentifier	0.172267	267
58	Census_OSBuildRevision	0.157988	235
54	Census_OSVersion	0.157986	307
36	Census_OEMNameIdentifier	0.146448	1620
8	DefaultBrowsersIdentifier	0.105794	557

	columns	skewness	unique
8	DefaultBrowsersIdentifier	0.105794	557
13	CountryIdentifier	0.044594	222
37	Census_OEMModelIdentifier	0.034577	40892
40	Census_ProcessorModelIdentifier	0.032664	2266
4	AvSigVersion	0.011614	6506
14	CityIdentifier	0.011183	37307
73	Census_FirmwareVersionIdentifier	0.010115	23544
44	Census_SystemVolumeTotalCapacity	0.005806	142066
0	MachineIdentifier	0.000002	499999

83 rows × 3 columns

In [9]:

```
for i in skew_data[skew_data.skewness >= 0.99]['columns'].values:
    drop_features.append(i)
drop_features = list(set(drop_features))
drop_features
```

Out[9]:

```
['SMode',
 'Census_IsWIMBootEnabled',
 'AutoSampleOptIn',
 'Census_IsPortableOperatingSystem',
 'PuaMode',
 'Census_IsVirtualDevice',
 'Census_DeviceFamily',
 'Census_IsFlightsDisabled',
 'Census_ProcessorClass',
 'Census_ThresholdOptIn',
 'UacLuaenable',
 'IsBeta',
 'Census_IsFlightingInternal']
```

We dropped features which have too many missing values or are too skewed. Also, we dropped MachineIdentifier column since every computer has a unique machine identifier.

In [10]:

```
#drop features
train.drop(drop_features, axis=1, inplace=True)
```

In [11]:

```
#drop MachineIdentifier
train.drop("MachineIdentifier",axis=1, inplace=True)
```

In [12]:

```
train.shape
```

Out[12]:

```
(499999, 69)
```

Now we reduced to 69 features (initially 83 features).

c) Checking Nan Values

In [13]:

```
#Check how many unique values each columns
#Nan Values
null_counts = train.isnull().sum()
null_counts = train.isnull().sum()
null_counts = null_counts / train.shape[0]
print(null_counts[null_counts != 0.0])
```

```
RtpStateBitfield                                     0.003748
DefaultBrowsersIdentifier                            0.951396
AVProductStatesIdentifier                            0.004062
AVProductsInstalled                                  0.004062
AVProductsEnabled                                    0.004062
CityIdentifier                                       0.036418
OrganizationIdentifier                               0.308717
GeoNameIdentifier                                    0.000006
OsBuildLab                                           0.000002
IsProtected                                          0.004040
IeVerIdentifier                                      0.006750
SmartScreen                                          0.356591
Firewall                                             0.010358
Census_OEMNameIdentifier                             0.010540
Census_OEMModelIdentifier                            0.011316
Census_ProcessorCoreCount                            0.004694
Census_ProcessorManufacturerIdentifier               0.004694
Census_ProcessorModelIdentifier                      0.004700
Census_PrimaryDiskTotalCapacity                      0.005932
Census_PrimaryDiskTypeName                           0.001490
Census_SystemVolumeTotalCapacity                     0.005932
Census_TotalPhysicalRAM                              0.009058
Census_ChassisTypeName                               0.000054
Census_InternalPrimaryDiagonalDisplaySizeInInches    0.005480
Census_InternalPrimaryDisplayResolutionHorizontal    0.005468
Census_InternalPrimaryDisplayResolutionVertical      0.005468
Census_PowerPlatformRoleName                         0.000004
Census_InternalBatteryType                           0.710283
Census_InternalBatteryNumberOfCharges                0.030250
Census_OSInstallLanguageIdentifier                   0.006636
Census_FirmwareManufacturerIdentifier                0.020520
Census_FirmwareVersionIdentifier                     0.017912
Census_IsAlwaysOnAlwaysConnectedCapable              0.007948
Wdft_IsGamer                                         0.034182
Wdft_RegionIdentifier                                0.034182
dtype: float64
```

If there are more than 10% of missing values, we manually replace those missing values.

In [14]:

```
null_counts[null_counts>=0.1]
```

Out[14]:

```
DefaultBrowsersIdentifier     0.951396
OrganizationIdentifier        0.308717
SmartScreen                   0.356591
Census_InternalBatteryType    0.710283
dtype: float64
```

In [15]:

```
train.DefaultBrowsersIdentifier.value_counts().unique()
```

Out[15]:

```
array([2571, 2356, 1574, 1308, 1177, 1030,  979,  811,  741,  715,  656,
        654,  605,  570,  432,  377,  352,  351,  294,  287,  270,  249,
        238,  221,  187,  155,  150,  129,  126,  117,  116,  114,  108,
        104,   91,   85,   84,   78,   76,   75,   68,   66,   62,   55,
         53,   52,   46,   44,   41,   40,   39,   37,   32,   31,   28,
         27,   26,   25,   24,   23,   22,   21,   20,   19,   18,   17,
         16,   15,   14,   13,   12,   11,   10,    9,    8,    7,    6,
          5,    4,    3,    2,    1], dtype=int64)
```

In [16]:

```
train.DefaultBrowsersIdentifier.fillna(0,inplace=True)
train.DefaultBrowsersIdentifier.fillna(0,inplace=True)
```

In [17]:

```
train.SmartScreen.value_counts()
```

Out[17]:

```
RequireAdmin    241671
ExistsNotSet     58779
Off              10458
Warn              7484
Prompt            1902
Block             1234
off                 81
On                  37
&#x01;              24
&#x02;              22
on                  11
OFF                  1
Name: SmartScreen, dtype: int64
```

In [18]:

```
import numpy as np
SmartScreen_dict = {
    'off': 'Off', '&#x02;': '2', '&#x01;': '1', 'on': 'On', 'requireadmin': 'RequireAdmin', 'OFF':
'Off',
    'Promt': 'Prompt', 'requireAdmin': 'RequireAdmin', 'prompt': 'Prompt', 'warn': 'Warn',
    '00000000': '0', '&#x03;': '3', np.nan: 'NoExist'
}
train.replace({'SmartScreen': SmartScreen_dict}, inplace=True)
print(train.SmartScreen.isnull().sum())
```

```
0
```

In [19]:

```
train.OrganizationIdentifier.value_counts()
```

Out[19]:

```
27.0    235447
18.0     98275
48.0      3613
50.0      2530
37.0      1109
11.0      1101
49.0       776
46.0       634
14.0       273
32.0       259
36.0       234
33.0       185
52.0       173
2.0        138
5.0        120
28.0        98
40.0        91
4.0         82
10.0        74
51.0        56
8.0         48
20.0        47
1.0         43
39.0        30
6.0         28
16.0        25
47.0        24
31.0        21
3.0         18
21.0        15
22.0        14
7.0         12
```

```
7.0         12
26.0        10
29.0         9
44.0         7
19.0         6
42.0         5
41.0         4
43.0         2
30.0         2
45.0         1
15.0         1
25.0         1
Name: OrganizationIdentifier, dtype: int64
```

In [20]:

```
train.replace({'OrganizationIdentifier': {np.nan: 0.0}}, inplace=True)
print(train.OrganizationIdentifier.isnull().sum())
```

```
0
```

In [21]:

```
train.Census_InternalBatteryType.value_counts()
```

Out[21]:

```
lion    113609
li-i     13782
#        10424
lip       3530
liio      1854
li p       448
li         371
nimh       256
real       148
pbac       127
bq20       120
vbox        86
unkn        22
lgi0        21
lipp        12
lipo        12
4cel         9
lhp0         6
batt         5
ithi         4
bad          3
ram          2
virt         2
ca48         1
lit          1
a140         1
asmb         1
 lio         1
Name: Census_InternalBatteryType, dtype: int64
```

In [22]:

```
census_bt_dict = {
    '˙˙˙': 'unknown', 'unkn': 'unknown', np.nan: 'unknown'
}
train.replace({'Census_InternalBatteryType': census_bt_dict}, inplace=True)
print(train.Census_InternalBatteryType.isnull().sum())
```

```
0
```

In [23]:

```
train['SmartScreen'] = train.SmartScreen.astype('category')
train['Census_InternalBatteryType'] = train.Census_InternalBatteryType.astype('category')
```

```
category_cols = train.select_dtypes(include='category').columns.tolist()
```

Now, Remove missing values from the train

In [24]:

```
train.dropna(inplace=True)
train.shape
```

Out[24]:

```
(429572, 69)
```

d) Select highly Correlated Features

First, we replaced the categorical values into numerical values

In [25]:

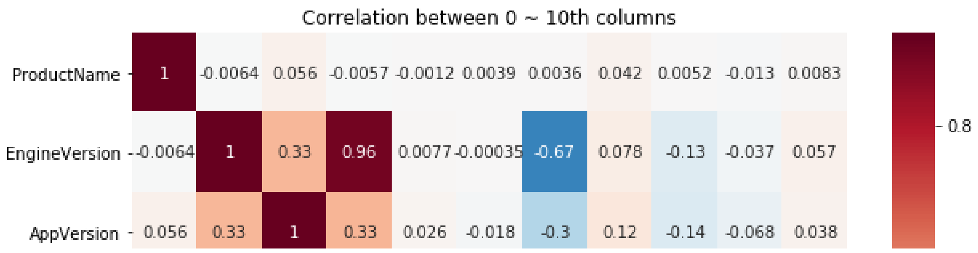
```
#Encode labels with value between 0 and n_classes-1.
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()

for col in category_cols:
    train[col] = le.fit_transform(train[col])
```

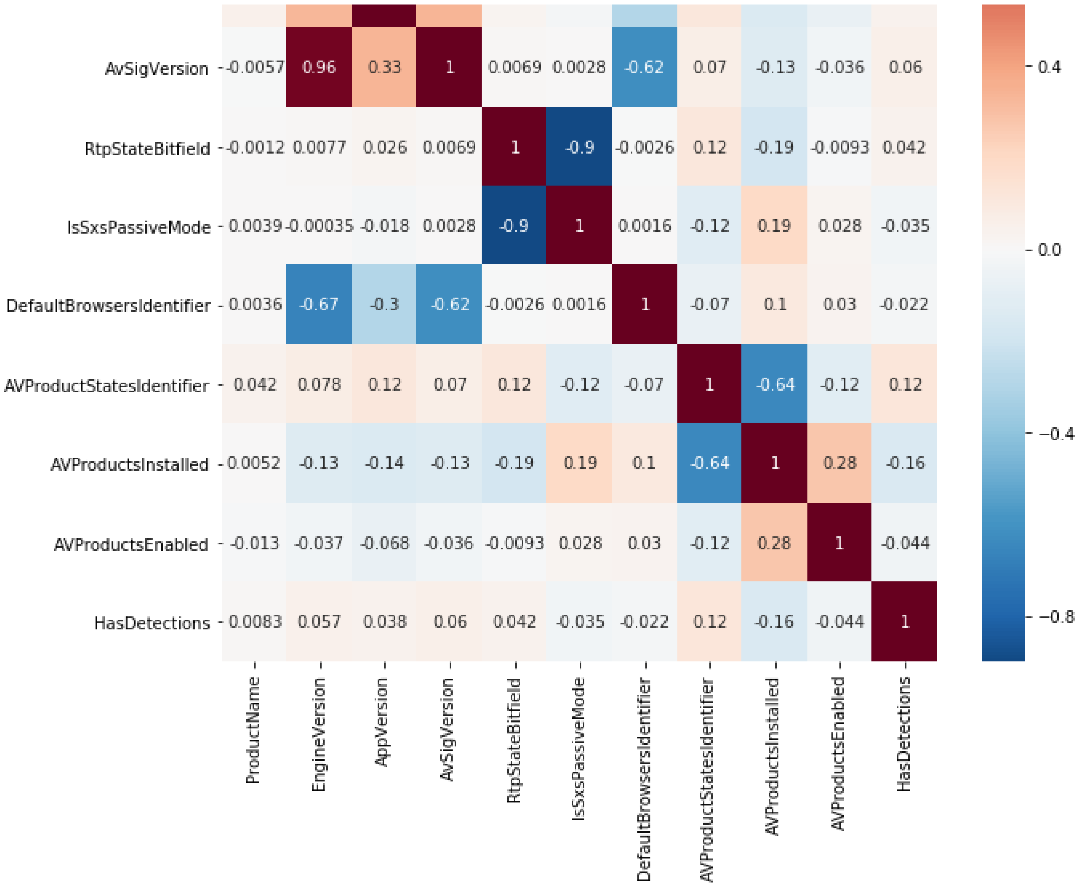
In [26]:

```
#Also, we implemented our own labelEncoder function.
#sklearn.preprocessing.LabelEncoder runs faster, so we used that library instead.
def myLabelEncode():
    for x in features:
        print(x)
        sample = train.loc[:, train.columns == x]
        sample = train.loc[:, train.columns == "ProductName"]

        for i,c in enumerate(sample[x].unique()):
            print(i,c)
            mask = (features.loc[:, features.columns == x] == c)
            sample[mask] = i

        sample = sample.astype(int)

        print(x, ": ", sample[x].value_counts().to_dict())
```

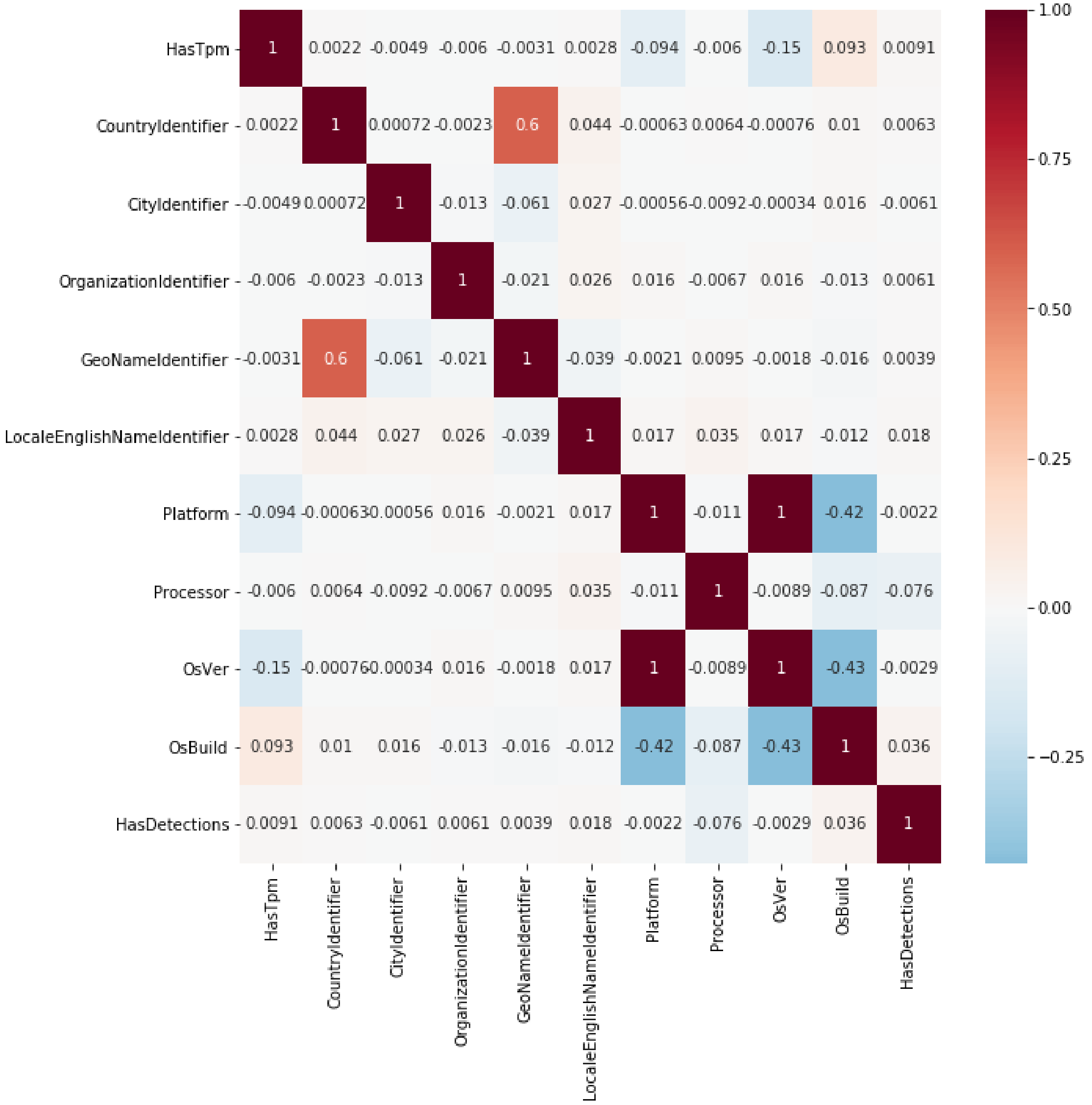
In [28]:

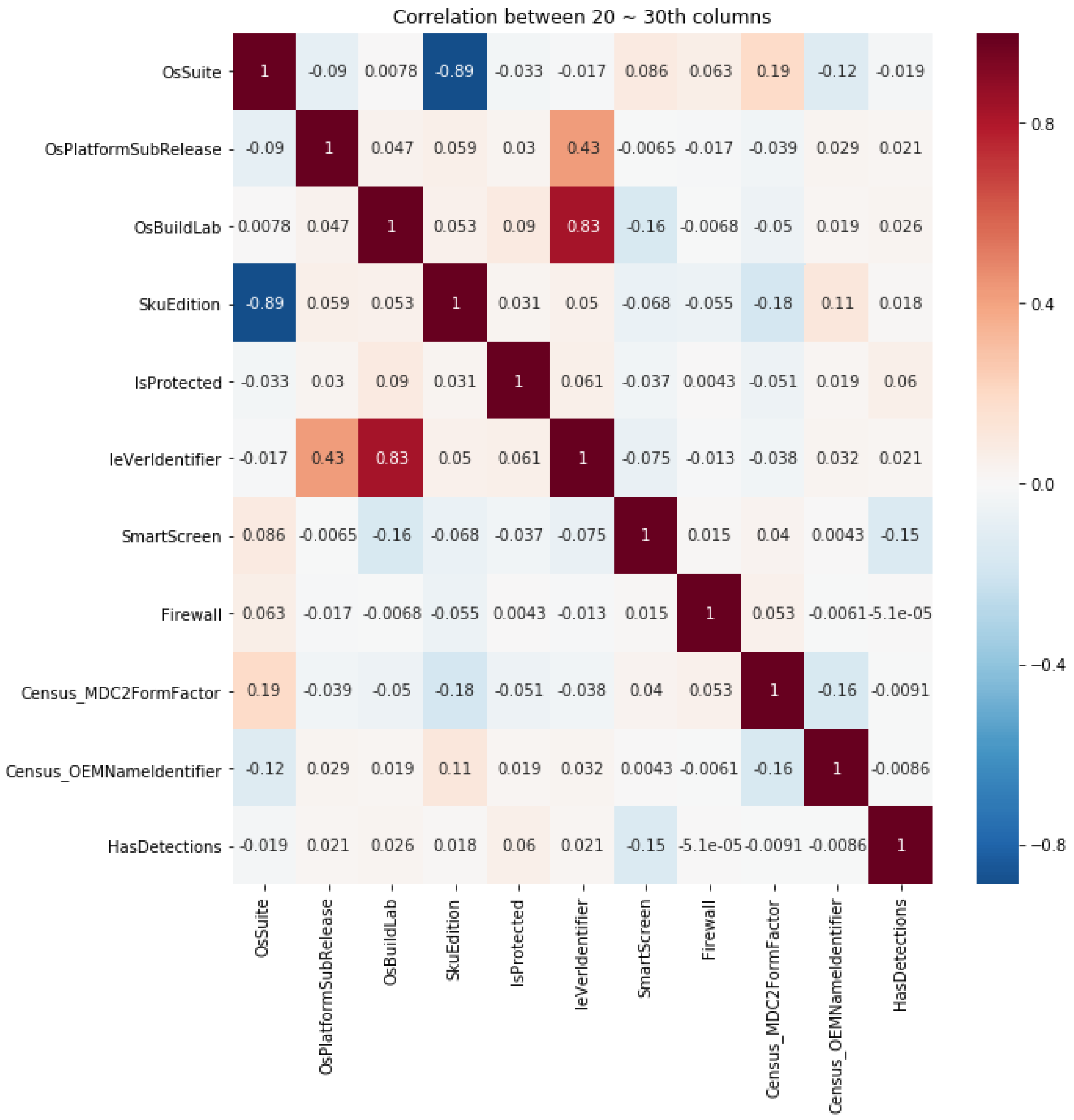
```
#Checking correations for each 10 columns
import seaborn as sns
import matplotlib.pyplot  as plt

cols = train.columns.tolist()
for i in range(0, len(cols), 10):
    plt.figure(figsize=(10,10))
    co_cols = cols[i:i+10]
    co_cols.append('HasDetections')
    sns.heatmap(train[co_cols].corr(), cmap='RdBu_r', annot=True, center=0.0)
    plt.title("Correlation between "+ str(i)+ " ~ " + str(i+10) + "th columns")
    plt.show()
```

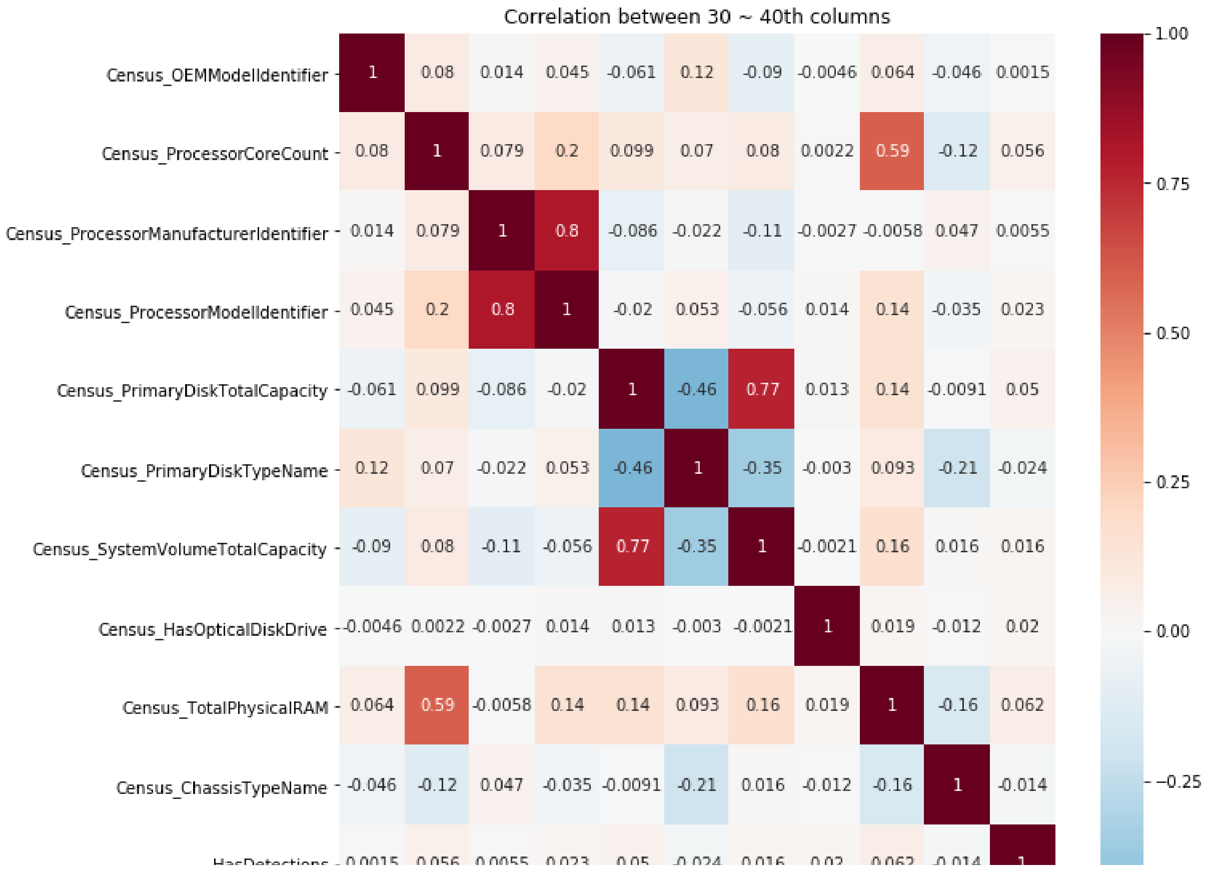




Correlation between 10 ~ 20th columns

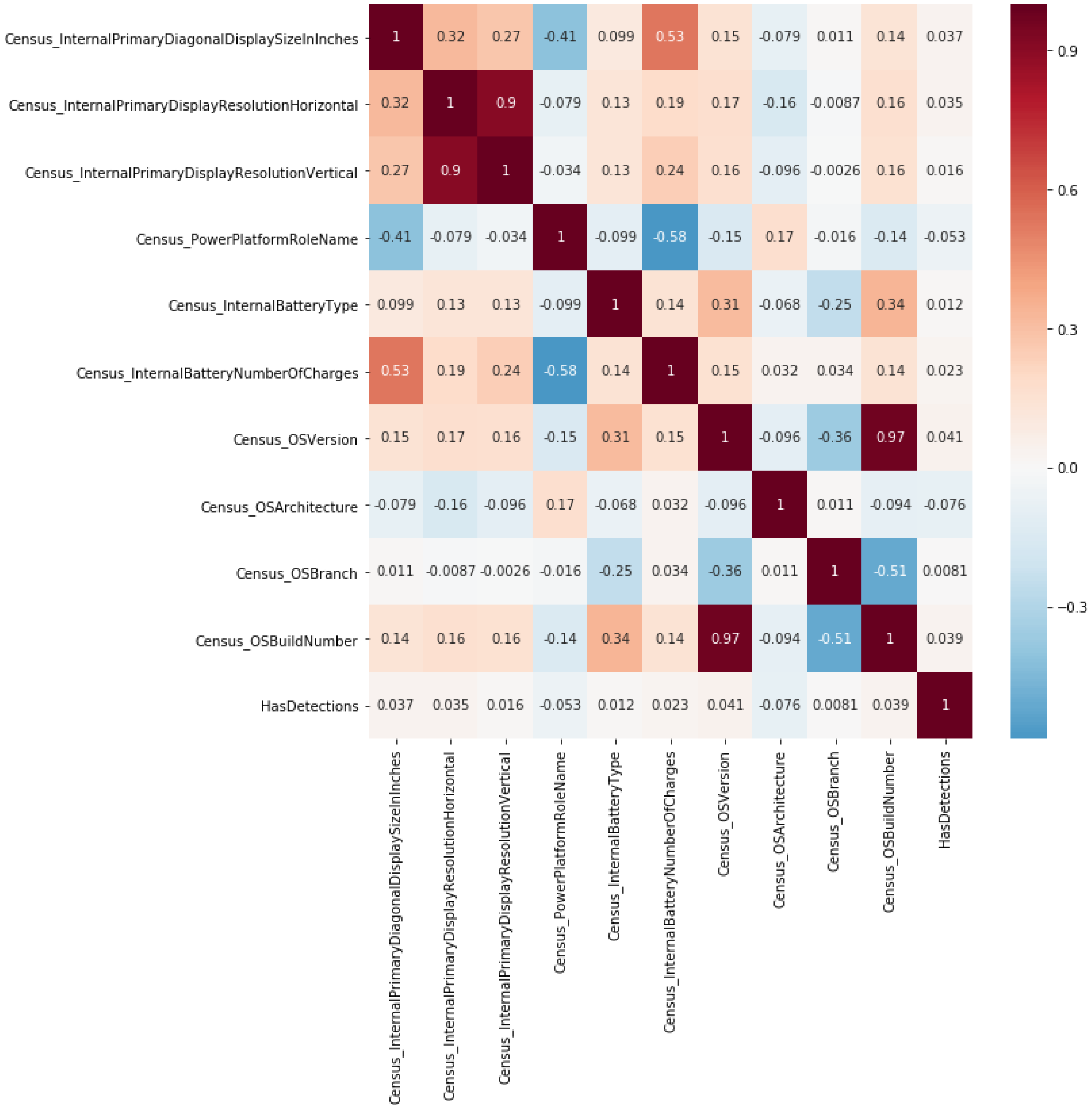



Correlation between 20 ~ 30th columns
OsSuite
OsPlatformSubRelease
OsBuildLab
SkuEdition
IsProtected
IeVerIdentifier
SmartScreen
Firewall
Census_MDC2FormFactor
Census_OEMNameIdentifier
HasDetections
1 -0.09 0.0078 -0.89 -0.033 -0.017 0.086 0.063 0.19 -0.12 -0.019
-0.09 1 0.047 0.059 0.03 0.43 -0.0065 -0.017 -0.039 0.029 0.021
0.0078 0.047 1 0.053 0.09 0.83 -0.16 -0.0068 -0.05 0.019 0.026
-0.89 0.059 0.053 1 0.031 0.05 -0.068 -0.055 -0.18 0.11 0.018
-0.033 0.03 0.09 0.031 1 0.061 -0.037 0.0043 -0.051 0.019 0.06
-0.017 0.43 0.83 0.05 0.061 1 -0.075 -0.013 -0.038 0.032 0.021
0.086 -0.0065 -0.16 -0.068 -0.037 -0.075 1 0.015 0.04 0.0043 -0.15
0.063 -0.017 -0.0068 -0.055 0.0043 -0.013 0.015 1 0.053 -0.0061 -5.1e-05
0.19 -0.039 -0.05 -0.18 -0.051 -0.038 0.04 0.053 1 -0.16 -0.0091
-0.12 0.029 0.019 0.11 0.019 0.032 0.0043 -0.0061 -0.16 1 -0.0086
-0.019 0.021 0.026 0.018 0.06 0.021 -0.15 -5.1e-05 -0.0091 -0.0086 1
0.8
0.4
0.0
−0.4
−0.8

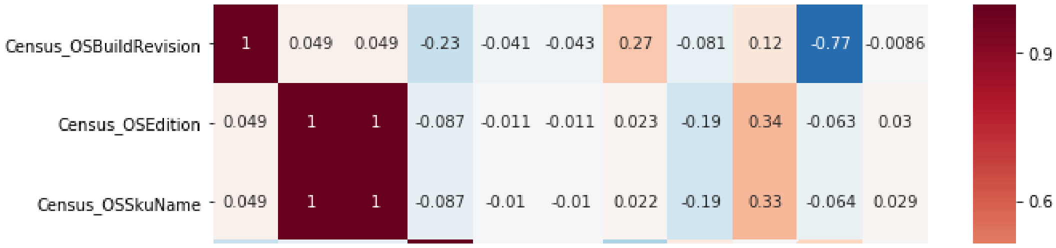

Correlation between 30 ~ 40th columns
Census_OEMModelIdentifier
Census_ProcessorCoreCount
Census_ProcessorManufacturerIdentifier
Census_ProcessorModelIdentifier
Census_PrimaryDiskTotalCapacity
Census_PrimaryDiskTypeName
Census_SystemVolumeTotalCapacity
Census_HasOpticalDiskDrive
Census_TotalPhysicalRAM
Census_ChassisTypeName
HasDetections
1 0.08 0.014 0.045 -0.061 0.12 -0.09 -0.0046 0.064 -0.046 0.0015
0.08 1 0.079 0.2 0.099 0.07 0.08 0.0022 0.59 -0.12 0.056
0.014 0.079 1 0.8 -0.086 -0.022 -0.11 -0.0027 -0.0058 0.047 0.0055
0.045 0.2 0.8 1 -0.02 0.053 -0.056 0.014 0.14 -0.035 0.023
-0.061 0.099 -0.086 -0.02 1 -0.46 0.77 0.013 0.14 -0.0091 0.05
0.12 0.07 -0.022 0.053 -0.46 1 -0.35 -0.003 0.093 -0.21 -0.024
-0.09 0.08 -0.11 -0.056 0.77 -0.35 1 -0.0021 0.16 0.016 0.016
-0.0046 0.0022 -0.0027 0.014 0.013 -0.003 -0.0021 1 0.019 -0.012 0.02
0.064 0.59 -0.0058 0.14 0.14 0.093 0.16 0.019 1 -0.16 0.062
-0.046 -0.12 0.047 -0.035 -0.0091 -0.21 0.016 -0.012 -0.16 1 -0.014
0.0015 0.056 0.0055 0.023 0.05 -0.024 0.016 0.02 0.062 -0.014 1
1.00
0.75
0.50
0.25
0.00
−0.25

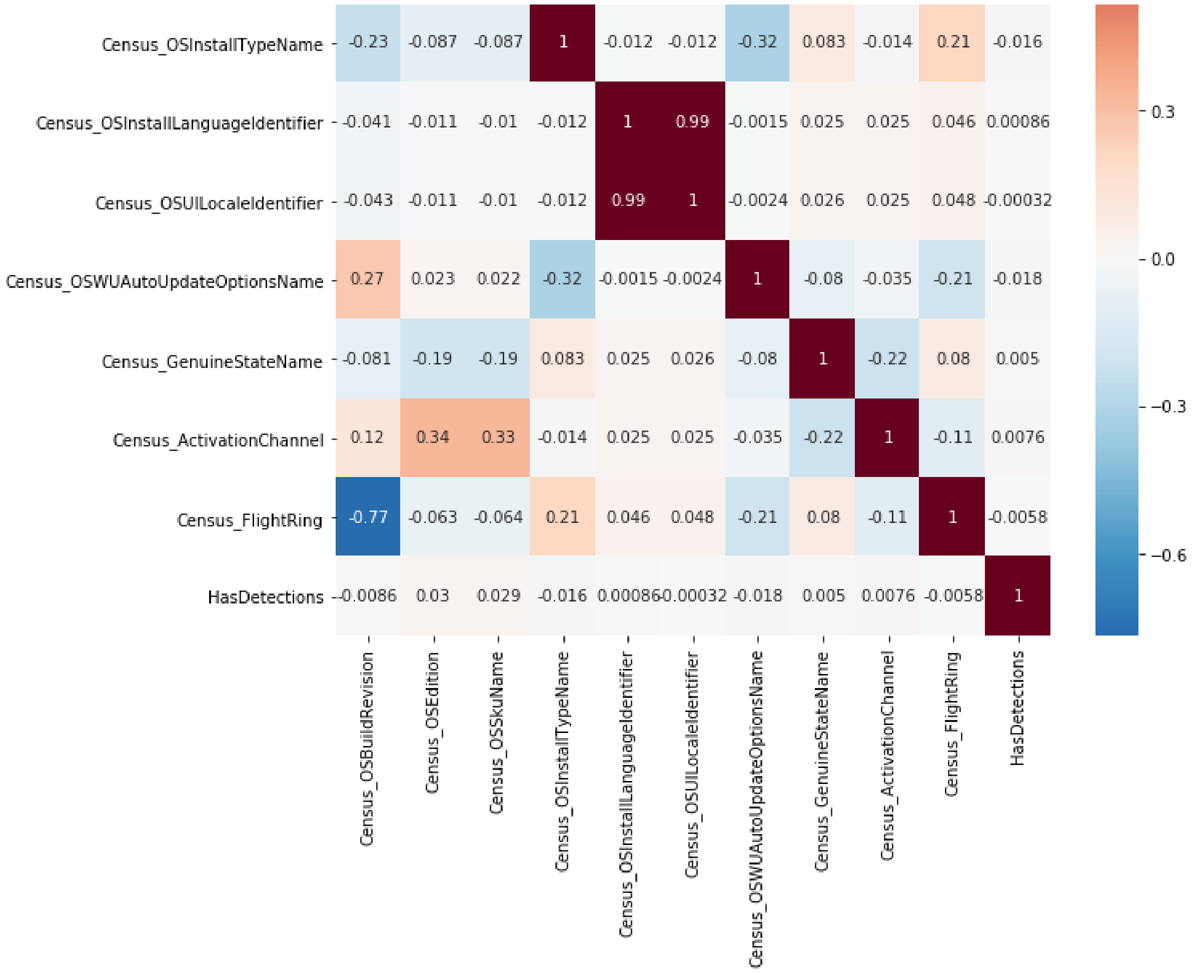
HasDetections
1
Census_OEMModelIdentifier
Census_ProcessorCoreCount
Census_ProcessorManufacturerIdentifier
Census_ProcessorModelIdentifier
Census_PrimaryDiskTotalCapacity
Census_PrimaryDiskTypeName
Census_SystemVolumeTotalCapacity
Census_HasOpticalDiskDrive
Census_TotalPhysicalRAM
Census_ChassisTypeName
HasDetections

Correlation between 40 ~ 50th columns

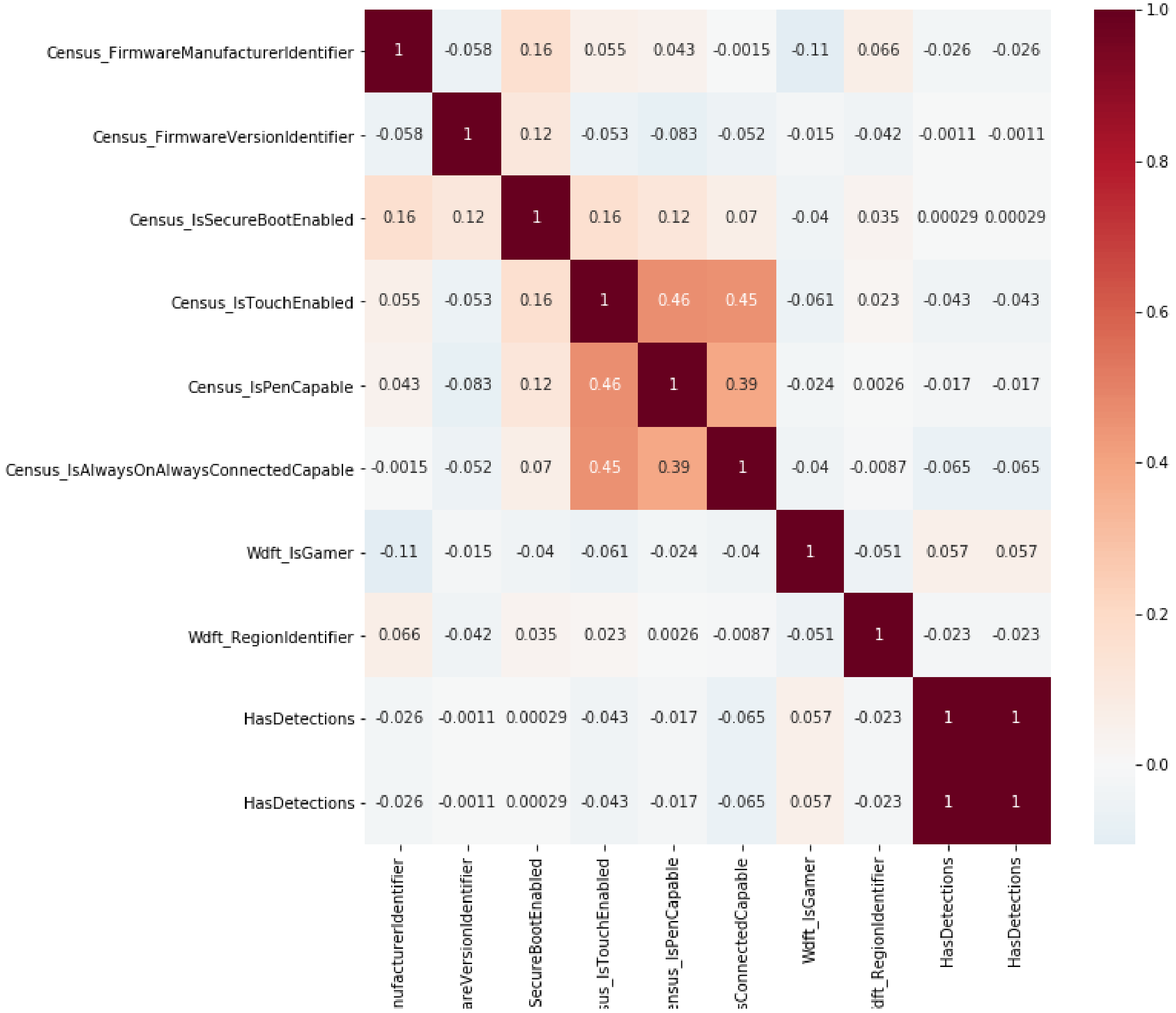


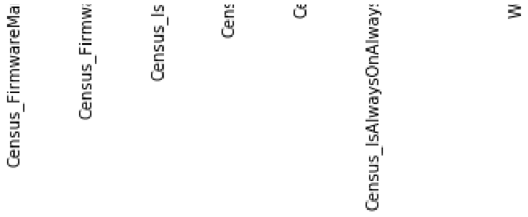
Correlation between 50 ~ 60th columns





Correlation between 60 ~ 70th columns

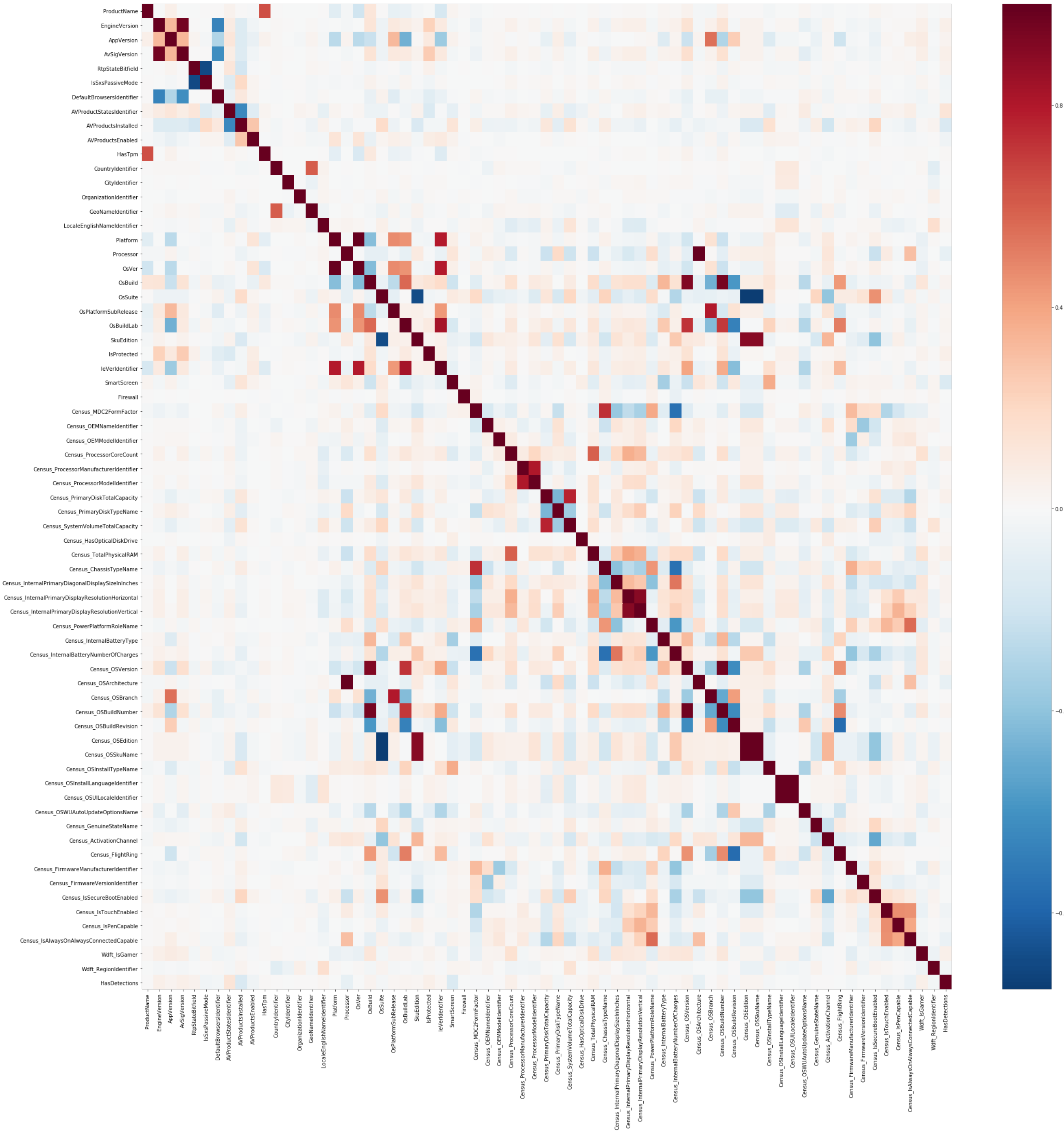




Also, we check the correlation for all columns

In [29]:

```
corr = train.corr()
high_corr = (corr >= 0.99).astype('uint8')
plt.figure(figsize=(35,35))
sns.heatmap(corr, cmap='RdBu_r', annot=False, center=0.0)
plt.show()
```



We could see there are high correlations (>0.99)

- OsVer vs Platform
- Census_OSUILocaleIdentifier vs Census_OSInstallLanguageIdentifier
- Census_OSArchitecture vs Processor
- Census_OSSkuName vs Census_OSEdition

Now, we check how many unique values each feature has and remove the one with less unique values

In [30]:

```
print("Unique values in OsVer: ", train.OsVer.nunique())
print("Unique values in Platform: ", train.Platform.nunique())
print()
print("Unique values in Census_OSUILocaleIdentifier: ", train.Census_OSUILocaleIdentifier.nunique(
))
print("Unique values in Census_OSInstallLanguageIdentifier: ",
train.Census_OSInstallLanguageIdentifier.nunique())
print()
print("Unique values in Census_OSArchitecture: ", train.Census_OSArchitecture.nunique())
print("Unique values in Processor: ", train.Processor.nunique())
print()
print("Unique values in Census_OSSkuName: ", train.Census_OSSkuName.nunique())
print("Unique values in Census_OSEdition: ", train.Census_OSEdition.nunique())
print()
```

```
Unique values in OsVer:  16
Unique values in Platform:  3

Unique values in Census_OSUILocaleIdentifier:  89
Unique values in Census_OSInstallLanguageIdentifier:  39

Unique values in Census_OSArchitecture:  3
Unique values in Processor:  3

Unique values in Census_OSSkuName:  16
Unique values in Census_OSEdition:  17
```

In [31]:

```
corr_remove = []
corr_remove.append('Platform')
corr_remove.append('Census_OSInstallLanguageIdentifier')
corr_remove.append('Census_OSArchitecture')
corr_remove.append('Census_OSSkuName')

train.drop(corr_remove,axis=1, inplace=True)
train.shape
```

Out[31]:

```
(429572, 65)
```

3. Generating train and test datasets

In [32]:

```
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.ensemble import AdaBoostClassifier
import matplotlib.pyplot as plt
from matplotlib.legend_handler import HandlerLine2D
from sklearn.metrics import roc_curve, auc
from sklearn.ensemble import BaggingClassifier
```

In [33]:

```
#This function generates train and test datsets.
#As a default, we used an 80/20 train/test split on the dataset
def train_test_generator(train , train_size=0.8, random_state=100):
```

```
def train_test_generator(train, train_size=0.8, random_state=100):
  features = train.loc[:, ~train.columns.isin(['HasDetections']) ]
  target = target = train.loc[:, train.columns == 'HasDetections']
  train_x, test_x, train_y, test_y = train_test_split(features, target, train_size=0.8,shuffle=True
,random_state=100)
  return train_x, test_x, train_y, test_y
```

In [34]:

```
train_x, test_x, train_y, test_y = train_test_generator(train)
```

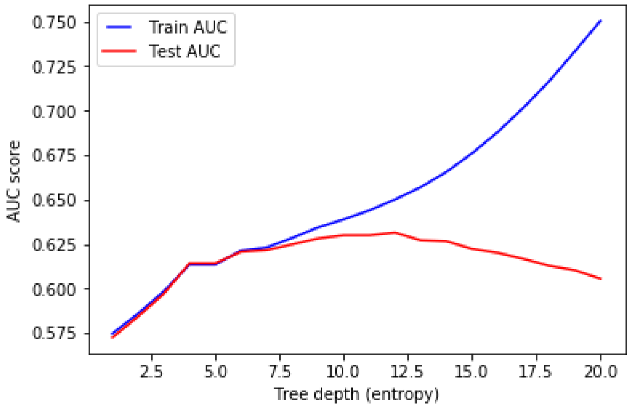
```
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\model_selection\_split.py:2179: FutureWarning:
From version 0.21, test_size will always complement train_size unless both are specified.
  FutureWarning)
```

4. Building Decision Tree, Bagging, Random Forest, AdaBoost models

a) Decision Tree

In [35]:

```
max_depths = np.linspace(1, 20, 20, endpoint=True)
train_results = []
test_results = []
for max_depth in max_depths:
    dt = DecisionTreeClassifier(criterion = 'entropy',max_depth=max_depth)
    dt.fit(train_x, train_y)
    train_pred = dt.predict(train_x)
    false_positive_rate, true_positive_rate, thresholds = roc_curve(train_y, train_pred)
    roc_auc = auc(false_positive_rate, true_positive_rate)
    # Add auc score to previous train results
    train_results.append(roc_auc)

    pred_y = dt.predict(test_x)
    false_positive_rate, true_positive_rate, thresholds = roc_curve(test_y, pred_y)
    roc_auc = auc(false_positive_rate, true_positive_rate)
    # Add auc score to previous test results
    test_results.append(roc_auc)

line1, = plt.plot(max_depths, train_results, 'b', label='Train AUC')
line2, = plt.plot(max_depths, test_results, 'r', label='Test AUC')
plt.legend(handler_map={line1: HandlerLine2D(numpoints=2)})
plt.ylabel('AUC score')
plt.xlabel('Tree depth (entropy)')
plt.show()
```



Tree depth 10 proves to be optimal.

In [36]:

```
from sklearn.model_selection import GridSearchCV
```

```
def DecisionTreeGridSearch(clf_tree=DecisionTreeClassifier(criterion="entropy")):
    param = {
        'max_depth': np.linspace(1, 20, 10, endpoint=False, dtype=int),
        'min_samples_split': np.linspace(0.1, 1.0, 5, endpoint=True)
    }
    # instantiate the grid
    grid = GridSearchCV(clf_tree, param, cv=5, scoring='accuracy')

    # fit the grid with data
    grid.fit(train_x, train_y)
    # summarize results
    print("Best: %f using %s" % (grid.best_score_, grid.best_params_))
```

In [37]:

```
clf = DecisionTreeClassifier(criterion="entropy", max_depth=10, random_state=100)
clf.fit(train_x, train_y)
pred_y = clf.predict(test_x)
clf_acur = accuracy_score(test_y, pred_y)
print("Accuracy:", clf_acur)
```

```
Accuracy: 0.6299249257987546
```

We implemented our own entropy and information gain function, but it ran very slow. Thus, to train a model, we used the sklearn library instead.

In [38]:

```
def entropy(target):
    elems, counts = np.unique(target, return_counts=1)
    entropy = np.sum([(-counts[i] / np.sum(counts))*np.log2(counts[i]/np.sum(counts)) for i in
range(len(elems))])

    return entropy

def IG(X, target):
    ig = [0.0] * train.shape[1]
    #calculate the entropy of the root node
    root_entropy = entropy(target)

    #calculate the target value and its corresponding counts
    for count, j in enumerate(X):
        xj = X[j]
        elems, counts = np.unique(xj, return_counts=1)
        print(elems, counts)
        split_entropy = 0
        for i in range(len(elems)):
            split_entropy += counts[i] / sum(counts) * entropy(target.iloc[[i for i in np.where(xj==
elems[i])[0]]])
        #calculate information gain
        IG = root_entropy - split_entropy
        ig[count] = IG
    return ig
```

b) Bagging

In [39]:

```
cart = clf = DecisionTreeClassifier(criterion = "entropy", max_depth=10, random_state = 100)
# param = {
#     'n_estimators': [1, 10, 25, 50, 100]
# }
# bag = BaggingClassifier(base_estimator=cart, random_state=100)

# # instantiate the grid
# grid = GridSearchCV(bag, param, cv=5, scoring='accuracy')

# # fit the grid with data
# grid.fit(train_x, train_y)
# # summarize results
# print("Best: %f using %s" % (grid.best_score_, grid.best_params_))
# print("Best: %f using %s" % (grid.best_score_, grid.best_params_))
```

In [40]:

```
bag = BaggingClassifier(base_estimator=cart, n_estimators=100, random_state=100)
bag.fit(train_x, train_y)
pred_y = bag.predict(test_x)
bag_acur = accuracy_score(test_y, pred_y)
print("Accuracy (Bagging):", bag_acur)
```

```
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\ensemble\bagging.py:621: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

```
Accuracy (Bagging): 0.6372461153465635
```

c) Random Forest

Fitting Random Forest

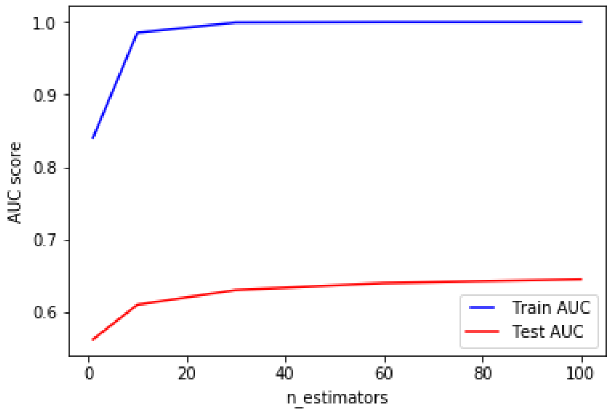
N_estimators: represents the number of trees in the forest

- the higher the number of trees, the better to learn the data
- However, more trees will slow down the training process

In [41]:

```
n_estimators = [1, 10, 30, 60, 100]
train_results = []
test_results = []
for estimator in n_estimators:
    rf = RandomForestClassifier(n_estimators=estimator, n_jobs=-1)
    rf.fit(train_x, train_y)
    train_pred = rf.predict(train_x)
    false_positive_rate, true_positive_rate, thresholds = roc_curve(train_y, train_pred)
    roc_auc = auc(false_positive_rate, true_positive_rate)
    train_results.append(roc_auc)
    pred_y = rf.predict(test_x)
    false_positive_rate, true_positive_rate, thresholds = roc_curve(test_y, pred_y)
    roc_auc = auc(false_positive_rate, true_positive_rate)
    test_results.append(roc_auc)

line1, = plt.plot(n_estimators, train_results, 'b', label='Train AUC')
line2, = plt.plot(n_estimators, test_results, 'r', label='Test AUC')
plt.legend(handler_map={line1: HandlerLine2D(numpoints=2)})
plt.ylabel('AUC score')
plt.xlabel('n_estimators')
plt.show()
```

```
C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().

C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().

C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().

C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().

C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().
```

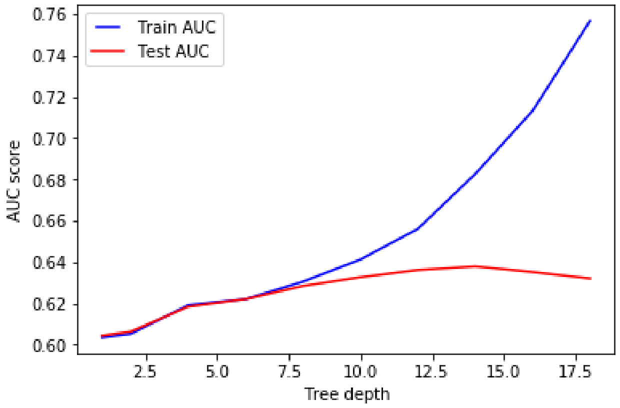


max_depth: The deeper the tree, the more splits it has and it captures more information about the data

In [43]:

```
max_depths = np.linspace(1, 20, 10, dtype = int, endpoint=False)
train_results = []
test_results = []
for max_depth in max_depths:
    rf = RandomForestClassifier(max_depth=max_depth, n_jobs=-1)
    rf.fit(train_x, train_y)
    train_pred = rf.predict(train_x)
    false_positive_rate, true_positive_rate, thresholds = roc_curve(train_y, train_pred)
    roc_auc = auc(false_positive_rate, true_positive_rate)
    train_results.append(roc_auc)
    y_pred = rf.predict(test_x)
    false_positive_rate, true_positive_rate, thresholds = roc_curve(test_y, y_pred)
    roc_auc = auc(false_positive_rate, true_positive_rate)
    test_results.append(roc_auc)
from matplotlib.legend_handler import HandlerLine2D
line1, = plt.plot(max_depths, train_results, 'b', label='Train AUC')
line2, = plt.plot(max_depths, test_results, 'r', label='Test AUC')
plt.legend(handler_map={line1: HandlerLine2D(numpoints=2)})
plt.ylabel('AUC score')
plt.xlabel('Tree depth')
plt.show()
```

```
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The defa
ult value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().

C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The defa
ult value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().

C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The defa
ult value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().

C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The defa
ult value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().

C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The defa
ult value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
ult value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().

C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The defa
ult value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().

C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The defa
ult value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().

C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The defa
ult value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().

C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The defa
ult value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().

C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The defa
ult value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().
```



depth: 10-12 would be optimal Otherwise, our model overfits for large depth values

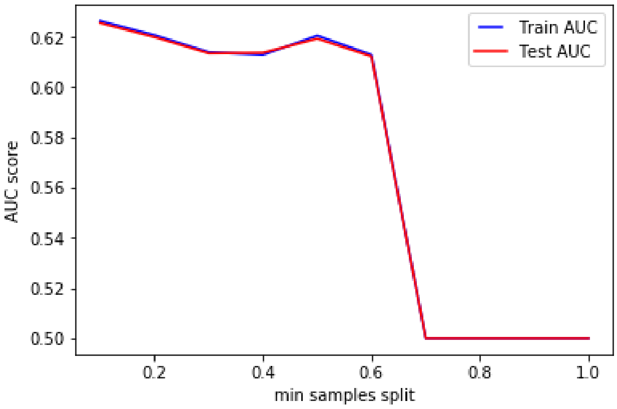
min_samples_split

In [44]:

```
min_samples_splits = np.linspace(0.1, 1.0, 10, endpoint=True)
train_results = []
test_results = []
for min_samples_split in min_samples_splits:
    rf = RandomForestClassifier(min_samples_split=min_samples_split)
    rf.fit(train_x, train_y)
    train_pred = rf.predict(train_x)
    false_positive_rate, true_positive_rate, thresholds = roc_curve(train_y, train_pred)
    roc_auc = auc(false_positive_rate, true_positive_rate)
    train_results.append(roc_auc)
    y_pred = rf.predict(test_x)
    false_positive_rate, true_positive_rate, thresholds = roc_curve(test_y, y_pred)
    roc_auc = auc(false_positive_rate, true_positive_rate)
    test_results.append(roc_auc)

line1, = plt.plot(min_samples_splits, train_results, 'b', label='Train AUC')
line2, = plt.plot(min_samples_splits, test_results, 'r', label='Test AUC')
plt.legend(handler_map={line1: HandlerLine2D(numpoints=2)})
plt.ylabel('AUC score')
plt.xlabel('min samples split')
plt.show()
```

```
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The defa
ult value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().

C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The defa
ult value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().

C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The defa
ult value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().

C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The defa
ult value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().

C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The defa
ult value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().

C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The defa
ult value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().

C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The defa
ult value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().

C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The defa
ult value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().

C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The defa
ult value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().

C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The defa
```

```
ult value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().
```



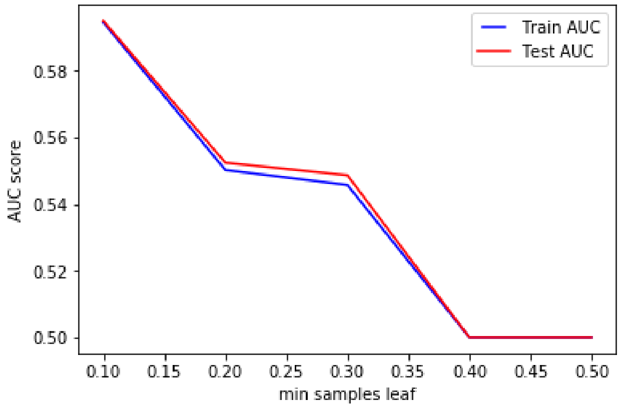
0.2 min samples split would be optimal. Increasing this values can cause underfitting

In [45]:

```
min_samples_leafs = np.linspace(0.1, 0.5, 5, endpoint=True)
train_results = []
test_results = []
for min_samples_leaf in min_samples_leafs:
    rf = RandomForestClassifier(min_samples_leaf=min_samples_leaf)
    rf.fit(train_x, train_y)
    train_pred = rf.predict(train_x)
    false_positive_rate, true_positive_rate, thresholds = roc_curve(train_y, train_pred)
    roc_auc = auc(false_positive_rate, true_positive_rate)
    train_results.append(roc_auc)
    y_pred = rf.predict(test_x)
    false_positive_rate, true_positive_rate, thresholds = roc_curve(test_y, y_pred)
    roc_auc = auc(false_positive_rate, true_positive_rate)
    test_results.append(roc_auc)
line1, = plt.plot(min_samples_leafs, train_results, 'b', label='Train AUC')
line2, = plt.plot(min_samples_leafs, test_results, 'r', label='Test AUC')
plt.legend(handler_map={line1: HandlerLine2D(numpoints=2)})
plt.ylabel('AUC score')
plt.xlabel('min samples leaf')
plt.show()
```

```
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The defa
ult value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().

C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The defa
ult value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().

C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The defa
ult value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().

C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The defa
ult value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
```

```
(n_samples,), for example using ravel().

C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The defa
ult value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().
```



Increasing this value can cause underfitting.

In [46]:

```
rf = RandomForestClassifier(n_estimators=100, random_state=100)
rf.fit(train_x, train_y)
pred_y = rf.predict(test_x)
rf_acur = accuracy_score(test_y, pred_y)
print("Accuracy:", rf_acur)
```

```
C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:2: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().
```

```
Accuracy: 0.643764185532212
```

d) AdaBoost

In [50]:

```
def AdaBoostGridSearch(Ada_clf=AdaBoostClassifier(base_estimator = cart)):
    param = [{
        'n_estimators': [1, 10, 50, 100],
        'learning_rate': [0.01, 0.05, 0.06, 0.1, 0.2, 1]
    }]
    # run grid search
    grid = GridSearchCV(Ada_clf, param, scoring='accuracy')
    grid.fit(train_x, train_y)
    # summarize results
    print("Best: %f using %s" % (grid.best_score_, grid.best_params_))

def learningAUC(learning_rate=[1, 0.5, 0.25, 0.1, 0.05, 0.01]):
    train_results = []
    test_results = []
    for eta in learning_rates:
        model = AdaBoostClassifier(base_estimator=cart, learning_rate=eta)
        model.fit(train_x, train_y)
        train_pred = model.predict(train_x)
        false_positive_rate, true_positive_rate, thresholds = roc_curve(train_y, train_pred)
        roc_auc = auc(false_positive_rate, true_positive_rate)
        train_results.append(roc_auc)
        y_pred = model.predict(test_x)
        false_positive_rate, true_positive_rate, thresholds = roc_curve(test_y, y_pred)
        roc_auc = auc(false_positive_rate, true_positive_rate)
        test_results.append(roc_auc)

    line1, = plt.plot(learning_rates, train_results, 'b', label='Train AUC')
    line2, = plt.plot(learning_rates, test_results, 'r', label='Test AUC')
    plt.legend(handler_map={line1: HandlerLine2D(numpoints=2)})
    plt.ylabel('AUC score')
    plt.xlabel('learning rate')
    plt.show()
```

In [51]:

```
Ada_clf = AdaBoostClassifier(base_estimator=cart, n_estimators=50, learning_rate=0.06,
random_state=100)
Ada_clf.fit(train_x, train_y)
pred_y = Ada_clf.predict(test_x)
Ada_acur = accuracy_score(test_y, pred_y)
print("Accuracy:", Ada_acur)
```

```
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

```
Accuracy: 0.6447418960600594
```

5. Results and Discussions

a) Feature Reduction Methologies

Log Regression with no feature reduction (base)

In [52]:

```
from sklearn.linear_model import LogisticRegression
clf = LogisticRegression().fit(train_x, train_y)
print(train_x.shape)
logReg = clf.score(test_x, test_y)
print("Log Regression with no feature reduction (base):", logReg)
```

```
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

```
(343657, 64)
Log Regression with no feature reduction (base): 0.5246813711226211
```

PCA best (maximum of 38, minimum of 10)

In [53]:

```
# n = 39 pca
copytrain_x = train_x.copy()
copytest_x = test_x.copy()

from sklearn.decomposition import PCA
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import Pipeline
from sklearn import linear_model, decomposition
from sklearn.pipeline import Pipeline
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import log_loss
```

```
logistic = linear_model.LogisticRegression()

pca = decomposition.PCA()
pipe = Pipeline(steps=[('pca', pca), ('logistic', logistic)])

# Plot the PCA spectrum
pca.fit(copytrain_x)

# Prediction
n_components = [10, 20, 30, 38]
Cs = np.logspace(-4, 4, 3)

# Parameters of pipelines can be set using '__' separated parameter names:
estimator = GridSearchCV(pipe,
                         dict(pca__n_components=n_components,
                              logistic__C=Cs))
estimator.fit(copytrain_x, train_y)
print(estimator.score(copytest_x,test_y))
```

```
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\model_selection\_split.py:2053: FutureWarning:
You should specify a value for 'cv' instead of relying on the default value. The default value wil
l change from 3 to 5 in version 0.22.
  warnings.warn(CV_WARNING, FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
```

```
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
```

```
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
```

```
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning: De
fault solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

```
0.5278589303381249
```

Correlation with target (54 features)

In [54]:

```
from scipy.stats import pearsonr
print("Correlations with 'HasDetections'")
dict_corr = dict()
cols = train.columns.tolist()
for i in cols:
    corr, _ = pearsonr(train[i], train['HasDetections'])
    dict_corr[i] = corr
print(dict_corr)

dropf = []
for k,v in dict_corr.items():
    if v == 0:
        dropf.append(k)
print(dropf)
newtrain_y = train_y.copy()
```

```
copytrain_x = train_x.copy()
copytest_x = test_x.copy()
copytrain_x.drop(dropf, axis=1, inplace=True)
copytest_x.drop(dropf, axis=1, inplace=True)
print(copytrain_x.shape)
clf = LogisticRegression(solver = 'lbfgs', ).fit(copytrain_x, train_y)
clf.score(copytest_x, test_y)
```

```
Correlations with 'HasDetections'
```

```
C:\Users\yamam\Anaconda3\lib\site-packages\numpy\core\fromnumeric.py:83: RuntimeWarning: overflow
encountered in reduce
  return ufunc.reduce(obj, axis, dtype, out, **passkwargs)
C:\Users\yamam\Anaconda3\lib\site-packages\scipy\stats\stats.py:5616: RuntimeWarning: overflow enc
ountered in multiply
  return np.sum(a*a, axis)
```

```
{'ProductName': 0.008267418086237035, 'EngineVersion': 0.05720826299704114, 'AppVersion':
0.038455368917377074, 'AvSigVersion': 0.05971006489962024, 'RtpStateBitfield': 0.0,
'IsSxsPassiveMode': -0.03501502744785843, 'DefaultBrowsersIdentifier': -0.022278523270603045,
'AVProductStatesIdentifier': 0.12408097897717965, 'AVProductsInstalled': -0.0,
'AVProductsEnabled': -0.04435794162918796, 'HasTpm': 0.00905932233646594, 'CountryIdentifier': 0.0
06307967150492906, 'CityIdentifier': -0.006059283132944537, 'OrganizationIdentifier': 0.0,
'GeoNameIdentifier': 0.0, 'LocaleEnglishNameIdentifier': 0.018168178838910372, 'Processor': -0.076
24611434347855, 'OsVer': -0.002898220751638023, 'OsBuild': 0.03584866153266568, 'OsSuite': -
0.019054384154674902, 'OsPlatformSubRelease': 0.02060705924085328, 'OsBuildLab':
0.026226641060607132, 'SkuEdition': 0.017504315152946478, 'IsProtected': 0.05954743671283125,
'IeVerIdentifier': 0.0, 'SmartScreen': -0.14628896725108953, 'Firewall': -5.135462940590799e-05, '
Census_MDC2FormFactor': -0.009080642256917017, 'Census_OEMNameIdentifier': -0.008646534317307513,
'Census_OEMModelIdentifier': 0.0014665149940282026, 'Census_ProcessorCoreCount': 0.0,
'Census_ProcessorManufacturerIdentifier': 0.0, 'Census_ProcessorModelIdentifier':
0.02316109499471877, 'Census_PrimaryDiskTotalCapacity': 0.04969273854761399,
'Census_PrimaryDiskTypeName': -0.024287844039813995, 'Census_SystemVolumeTotalCapacity':
0.016053763125566434, 'Census_HasOpticalDiskDrive': 0.019957428162376847,
'Census_TotalPhysicalRAM': 0.06193583013939577, 'Census_ChassisTypeName': -0.014213401772054242, '
Census_InternalPrimaryDiagonalDisplaySizeInInches': 0.036988125235285045,
'Census_InternalPrimaryDisplayResolutionHorizontal': 0.035134915815773715,
'Census_InternalPrimaryDisplayResolutionVertical': 0.016283893319071492,
'Census_PowerPlatformRoleName': -0.052797415542474864, 'Census_InternalBatteryType':
0.012370029273438173, 'Census_InternalBatteryNumberOfCharges': 0.022988762099357414,
'Census_OSVersion': 0.04127424950026324, 'Census_OSBranch': 0.00805959855082222,
'Census_OSBuildNumber': 0.03903575635698735‌4, 'Census_OSBuildRevision': -0.008566500866985856, 'Ce
nsus_OSEdition': 0.029570971610502362, 'Census_OSInstallTypeName': -0.016418661495050415,
'Census_OSUILocaleIdentifier': -0.00032221781817971167, 'Census_OSWUAutoUpdateOptionsName': -0.017
912593994587347, 'Census_GenuineStateName': 0.005036311360976625, 'Census_ActivationChannel': 0.00
7582295912598923, 'Census_FlightRing': -0.005778564188307404,
'Census_FirmwareManufacturerIdentifier': -0.0, 'Census_FirmwareVersionIdentifier': -
0.0010508910356106663, 'Census_IsSecureBootEnabled': 0.00029458618767595014,
'Census_IsTouchEnabled': -0.04296028225839951, 'Census_IsPenCapable': -0.017070424203206216,
'Census_IsAlwaysOnAlwaysConnectedCapable': -0.06528627236844896, 'Wdft_IsGamer': 0.0,
'Wdft_RegionIdentifier': -0.0, 'HasDetections': 1.0}
['RtpStateBitfield', 'AVProductsInstalled', 'OrganizationIdentifier', 'GeoNameIdentifier',
'IeVerIdentifier', 'Census_ProcessorCoreCount', 'Census_ProcessorManufacturerIdentifier',
'Census_FirmwareManufacturerIdentifier', 'Wdft_IsGamer', 'Wdft_RegionIdentifier']
(343657, 54)
```

```
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

Out[54]:

```
0.5050689635104464
```

Use AdaBoost with default hyperparameters

In [58]:

```
Ada_default = AdaBoostClassifier(n_estimators=100, learning_rate=1)
Ada_default.fit(train_x, train_y)
print(Ada_default.feature_importances_)
pred_y = Ada_default.predict(test_x)
```

```
print("Accuracy (AdaBoost):", accuracy_score(test_y, pred_y))
```

```
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

```
[0.   0.   0.08 0.08 0.02 0.01 0.02 0.11 0.02 0.01 0.   0.04 0.   0.
 0.01 0.03 0.   0.   0.   0.01 0.   0.01 0.   0.   0.   0.05 0.   0.
 0.02 0.   0.02 0.   0.01 0.03 0.   0.04 0.01 0.06 0.01 0.04 0.   0.01
 0.01 0.   0.   0.01 0.01 0.   0.03 0.02 0.01 0.04 0.   0.02 0.01 0.
 0.04 0.   0.   0.   0.   0.   0.02 0.03]
Accuracy (AdaBoost): 0.6395623581446779
```

In [59]:

```
from sklearn.utils.validation import column_or_1d

zero_feature = [ c for c, i in enumerate(Ada_default.feature_importances_) if i == 0.0]
use_feature = [c for c, i in enumerate(Ada_default.feature_importances_) if i != 0.0]
feature1 = train.iloc[:, use_feature]
zero_col = train.iloc[:,zero_feature].axes[1]
traindrop_x, testdrop_x, traindrop_y, testdrop_y = train_test_generator(feature1)
print(traindrop_x.shape)
clf = LogisticRegression(solver = 'lbfgs', ).fit(traindrop_x, train_y)
clf.score(testdrop_x, column_or_1d(test_y, warn =True))
```

```
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\model_selection\_split.py:2179: FutureWarning:
From version 0.21, test_size will always complement train_size unless both are specified.
  FutureWarning)
```

```
(343657, 36)
```

```
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:758:
ConvergenceWarning: lbfgs failed to converge. Increase the number of iterations.
  "of iterations.", ConvergenceWarning)
C:\Users\yamam\Anaconda3\lib\site-packages\ipykernel_launcher.py:10: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  # Remove the CWD from sys.path while we load stuff.
```

Out[59]:

```
0.5566664726764826
```

Model Performances vs. Number of Trees

In [ ]:

```
import matplotlib as mpl
from sklearn import model_selection
#Decision Tree
clf = DecisionTreeClassifier(criterion = "entropy", max_depth=10, random_state = 100)
clf.fit(train_x, train_y)
dt_train_err = 1.0 - clf.score(train_x,train_y)
pred_y = clf.predict(test_x)
dt_err = 1.0 - accuracy_score(test_y, pred_y)

n_estimators = [1, 10, 25, 50, 75, 100]
plt.figure(figsize=(8, 8))
plt.title('Decision Tree, Bagging, Random Forest and AdaBoost comparison')
plt.plot([0,100], [dt_err] * 2, 'k-', color="black", Label="Decision Tree test")
plt.plot([0,100], [dt_train_err] * 2, 'k--', color="black", Label="Decision Tree train")

bagging_err = np.zeros(6)
rf_err = np.zeros(6)
```

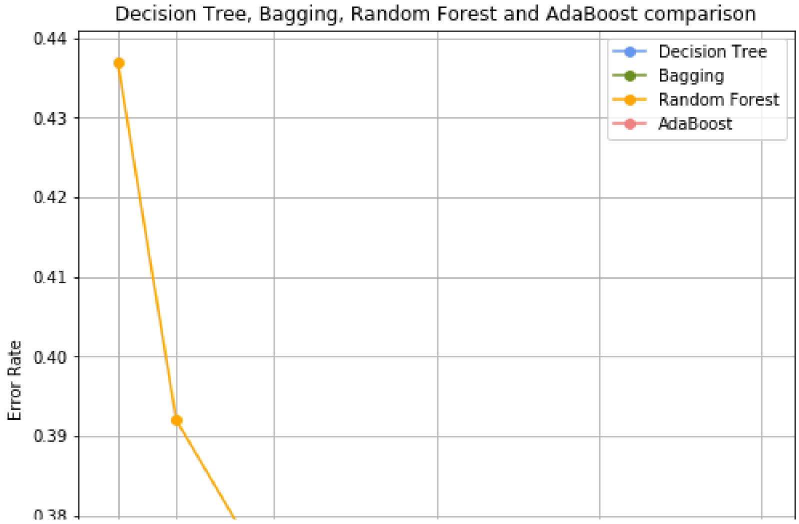
```
ada_err = np.zeros(6)

#Bagging
for c, i in enumerate(n_estimators):
    seed=7
    bag = BaggingClassifier(base_estimator=clf, n_estimators=i, random_state=seed)
    bag.fit(train_x, train_y)
    pred_y = bag.predict(test_x)
    bagging_err[c] = 1.0 - accuracy_score(test_y, pred_y)
    print("Accuracy bagging:", accuracy_score(test_y, pred_y))

    rf_clf = RandomForestClassifier(n_estimators=i, random_state=100)
    rf_clf.fit(train_x, train_y)
    pred_y = rf_clf.predict(test_x)
    rf_err[c] = 1.0 - accuracy_score(test_y, pred_y)
    print("Accuracy random forest:", accuracy_score(test_y, pred_y))

    ada_clf = AdaBoostClassifier(base_estimator=clf,n_estimators=i, learning_rate=0.06, random_stat
e=100)
    ada_clf.fit(train_x, train_y)
    pred_y = ada_clf.predict(test_x)
    ada_err[c] = 1.0 - accuracy_score(test_y, pred_y)
    print("Accuracy AdaBoost:", accuracy_score(test_y, pred_y))
```

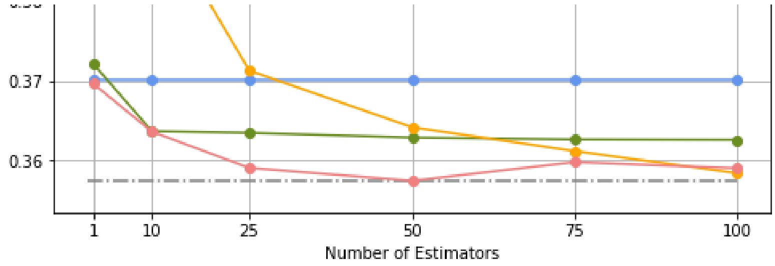
In [69]:

```
#As a result, we got the following
dt_err = 0.3700750742012454
bagging_err = [0.37214715, 0.3636711,  0.36346267, 0.36287213, 0.36262896, 0.36258265]
rf_err = [0.43696808, 0.39199407, 0.3712787,  0.36416901, 0.36116997, 0.35839094]
ada_err = [0.36964602, 0.36361321, 0.35905096, 0.35745302, 0.35978046, 0.35901622]
```

In [70]:

```
from mpl_toolkits.axes_grid1.inset_locator import zoomed_inset_axes
from mpl_toolkits.axes_grid1.inset_locator import mark_inset
from mpl_toolkits.axes_grid1.inset_locator import inset_axes
n_estimators = [1, 10, 25, 50, 75, 100]
plt.figure(figsize=(8, 8))
plt.title('Decision Tree, Bagging, Random Forest and AdaBoost comparison')
plt.plot(n_estimators, [dt_err] * 6, 'o-', color="cornflowerblue", label="Decision Tree")
plt.plot(n_estimators, bagging_err, 'o-', color="olivedrab", label='Bagging')
plt.plot(n_estimators, rf_err, 'o-', color="orange", label='Random Forest')
plt.plot(n_estimators, ada_err, 'o-', color="lightcoral", label='AdaBoost')
plt.legend(loc='best')
plt.xlabel('Number of Estimators')
plt.ylabel('Error Rate')
plt.xticks(n_estimators)
plt.grid(b=None)
plt.hlines(ada_err[3],0,100, linestyles="dashdot",color="gray")
# this is an inset axes over the main axes
plt.legend(loc='best')
plt.savefig('comparison1.png')
plt.show()
```





In [71]:

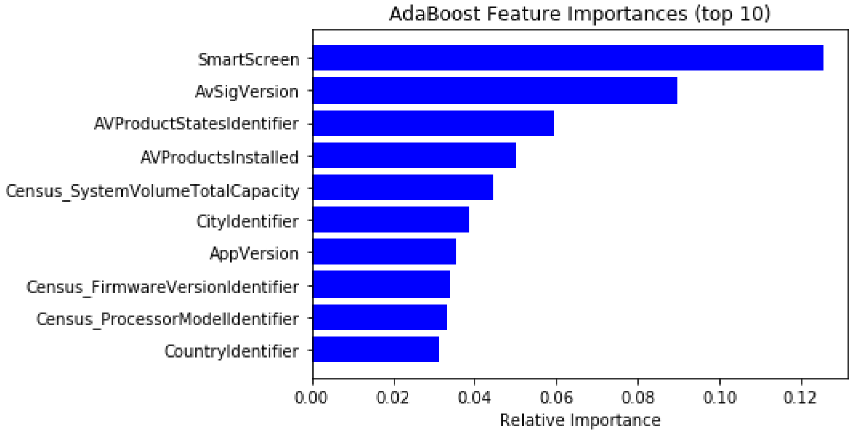
```
print("Final Results")
d = {"accuracy" : pd.Series([clf_acur, bag_acur, rf_acur, Ada_acur],
                            index=["Decision Tree", "Bagging", "Random Forest", "AdaBoost"])}
df = pd.DataFrame(d)
print(df)
```

```
Final Results
               accuracy
Decision Tree  0.629925
Bagging        0.637246
Random Forest  0.643764
AdaBoost       0.644742
```

Comparing Feature Importances

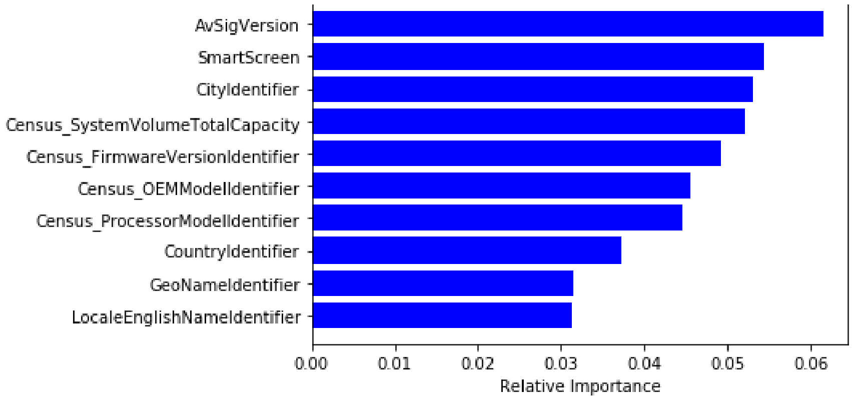
In [73]:

```
features = train.loc[:, train.columns != 'HasDetections'].columns

ada_importances = Ada_clf.feature_importances_
ada_indices = np.argsort(ada_importances)[-10:]   # top 10 features
plt.title('AdaBoost Feature Importances (top 10)')
plt.barh(range(len(ada_indices)), ada_importances[ada_indices], color='b', align='center')
plt.yticks(range(len(ada_indices)), [features[i] for i in ada_indices])
plt.xlabel('Relative Importance')
plt.show()
```



In [74]:

```
features = train.loc[:, train.columns != 'HasDetections'].columns

rf_importances = rf.feature_importances_
rf_indices = np.argsort(rf_importances)[-10:]   # top 10 features
plt.title('Random Forest Feature Importances (top 10)')
plt.barh(range(len(rf_indices)), rf_importances[rf_indices], color='b', align='center')
plt.yticks(range(len(rf_indices)), [features[i] for i in rf_indices])
plt.xlabel('Relative Importance')
plt.show()
```



a) Apply Logistic Regression with the top 10 features

In [75]:

```
ad_important = [ada_importances[i] for i in ada_indices]
feature_top = [features[i] for i in ada_indices]
dropf = [i for i in features if i not in feature_top]

copytrain_x = train_x.copy()
copytest_x = test_x.copy()
copytrain_x.drop(dropf, axis=1, inplace=True)
copytest_x.drop(dropf, axis=1, inplace=True)

from sklearn.linear_model import LogisticRegression
lg = LogisticRegression(solver = 'lbfgs', ).fit(copytrain_x, train_y)
ada_log = lg.score(copytest_x, test_y)
print(ada_log)
```

```
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

```
0.5436070534830938
```

In [76]:

```
rf_important = [rf_importances[i] for i in rf_indices]
feature_top = [features[i] for i in rf_indices]
dropf = [i for i in features if i not in feature_top]

copytrain_x1 = train_x.copy()
copytest_x1 = test_x.copy()
copytrain_x1.drop(dropf, axis=1, inplace=True)
copytest_x1.drop(dropf, axis=1, inplace=True)

lg = LogisticRegression(solver = 'lbfgs', ).fit(copytrain_x1, train_y)
rf_log = lg.score(copytest_x1, test_y)
print(rf_log)
```

```
C:\Users\yamam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

```
0.5155211546295757
```

In [77]:

```
print("Comparing Accuracy with the Top 10 Featuress")
d = {"accuracy" : pd.Series([ada_log, rf_log],
                            index=["AdaBoost", "Random Forest", ])}
df = pd.DataFrame(d)
print(df)
```

```
Comparing Accuracy with the Top 10 Featuress
               accuracy
AdaBoost       0.543607
Random Forest  0.515521
```