other_models

August 5, 2018

1 Prediction model tuning

The prediction models were further tuned on the selected data framework. This included methods for dealing with imbalanced data, such as upsampling the minor class using the `imbalanced-learn` package.

In [3]:
```
import pandas as pd
import numpy as np
import dill

from imblearn.pipeline import make_pipeline
from imblearn.ensemble import EasyEnsemble, BalanceCascade
from imblearn.over_sampling import ADASYN
from imblearn.under_sampling import EditedNearestNeighbours

from sklearn.pipeline import FeatureUnion
from sklearn.preprocessing import Normalizer
from sklearn.cross_validation import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier, VotingClassifi
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC, LinearSVC
from sklearn.grid_search import GridSearchCV
from sklearn.metrics import confusion_matrix, f1_score, roc_auc_score, roc_curve
from sklearn.metrics import auc, accuracy_score, precision_recall_curve
from sklearn.metrics import precision_score, recall_score
from sklearn.base import BaseEstimator, ClassifierMixin

from scipy.stats import mode

%matplotlib inline
import matplotlib.pyplot as plt

import seaborn as sns

sns.set_context('talk')
sns.set_palette('dark')
sns.set_style('white')
```

```
/home/abhijit/anaconda3/envs/idp/lib/python3.6/site-packages/sklearn/cross_validation.py:41: De
  "This module will be removed in 0.20.", DeprecationWarning)
/home/abhijit/anaconda3/envs/idp/lib/python3.6/site-packages/sklearn/grid_search.py:42: Deprec
  DeprecationWarning)
```

```
In [4]: class MYVC(BaseEstimator, ClassifierMixin):
            def __init__(self, classifiers=None):
                self.classifiers = classifiers

            def fit(self, X, y):
                for classifier in self.classifiers:
                    classifier.fit(X, y)

            def predict_proba(self, X):
                self.predictions_ = list()
                for classifier in self.classifiers:
                    self.predictions_.append(classifier.predict_proba(X))
                return np.mean(self.predictions_, axis=0)

            def predict(self, X):
                self.predictions_ = list()
                for classifier in self.classifiers:
                    self.predictions_.append(classifier.predict(X))
                return mode(self.predictions_, axis=0).mode[0]
```

1.1 Combine data

First import and normalize the features.

```
In [35]: with open('./pkl/11_features_engineered.pkl', 'rb') as fh:
             features = dill.load(fh)
         features.head()
         # features = features.drop(['index', 'country'], axis=1).T.groupby(level=0).first().T
         features['date'] = pd.to_datetime(features['date'])

         for col in features.columns:
             if col not in ['date', 'location']:
                 features[col] = features[col].astype(np.float)
```

```
In [38]: feat_cols = [x for x in features.columns if x not in ['date','location']]
         features[feat_cols] = Normalizer().fit_transform(features[feat_cols])
```

Import the predictors.

```
In [84]: #framework_a_first = pd.read_pickle('../pkl/10_class_balancing_framework_a_first.pkl'
         framework_a_first = pd.read_pickle('./pkl/10_class_balancing_fwf.pkl')
```

```
In [85]: print (framework_a_first.shape, framework_a_first.isnull().sum().max())

         fwd_a_first = pd.merge(framework_a_first,
                                features,
                                on=['date','location'], how='left').dropna()

         print (fwd_a_first.shape, fwd_a_first.isnull().sum().max())

         print (fwd_a_first.zika_bool.value_counts())
```

```
(1605, 5) 0
(1213, 29) 0
1    1004
0     209
Name: zika_bool, dtype: int64
```

Split the data into train and test portions.

```
In [86]: train, test = train_test_split(fwd_a_first, test_size=0.30, random_state=42)

         adasyn_3 = ADASYN(k=3)
         X_resampled_3, Y_resampled_3 = adasyn_3.fit_sample(train[feat_cols], train['zika_bool

         adasyn_5 = ADASYN(k=5)
         X_resampled_5, Y_resampled_5 = adasyn_5.fit_sample(train[feat_cols], train['zika_bool

         adasyn_7 = ADASYN(k=7)
         X_resampled_7, Y_resampled_7 = adasyn_7.fit_sample(train[feat_cols], train['zika_bool

         enn_3 = EditedNearestNeighbours(size_ngh=3)
         X_dwnresampled_3, Y_dwnresampled_3 = enn_3.fit_sample(train[feat_cols], train['zika_bo

         enn_5 = EditedNearestNeighbours(size_ngh=5)
         X_dwnresampled_5, Y_dwnresampled_5 = enn_5.fit_sample(train[feat_cols], train['zika_bo

         enn_7 = EditedNearestNeighbours(size_ngh=7)
         X_dwnresampled_7, Y_dwnresampled_7 = enn_7.fit_sample(train[feat_cols], train['zika_bo

         Ytest = test['zika_bool']
         Xtest = test[feat_cols+['date','location']]
```

```
/home/abhijit/.local/lib/python3.6/site-packages/imblearn/utils/deprecation.py:50: DeprecationW
  category=DeprecationWarning)
/home/abhijit/.local/lib/python3.6/site-packages/imblearn/utils/deprecation.py:50: DeprecationW
  category=DeprecationWarning)
/home/abhijit/.local/lib/python3.6/site-packages/imblearn/utils/deprecation.py:50: DeprecationW
  category=DeprecationWarning)
/home/abhijit/.local/lib/python3.6/site-packages/imblearn/utils/deprecation.py:50: DeprecationW
```

```
  category=DeprecationWarning)
/home/abhijit/.local/lib/python3.6/site-packages/imblearn/utils/deprecation.py:50: DeprecationW
  category=DeprecationWarning)
/home/abhijit/.local/lib/python3.6/site-packages/imblearn/utils/deprecation.py:50: DeprecationW
  category=DeprecationWarning)
```

```
In [87]: test
```

Out[87]:

	location	cases_first_date	\
609	Colombia-Cordoba-San_Jose_De_Ure	0	
1192	Colombia-Valle_Del_Cauca-Dagua	0	
46	Brazil-Rio_de_Janeiro	229	
170	Colombia-Antioquia-Segovia	0	
781	Colombia-Huila-Nataga	0	
890	Colombia-Narino-La_Cruz	0	
337	Colombia-Boyaca-Jerico	0	
905	Colombia-Narino-Policarpa	0	
135	Colombia-Antioquia-Montebello	0	
124	Colombia-Antioquia-Ituango	0	
937	Colombia-Norte_Santander-Cucutilla	0	
140	Colombia-Antioquia-Necocli	0	
896	Colombia-Narino-Linares	0	
670	Colombia-Cundinamarca-Madrid	0	
162	Colombia-Antioquia-San_Pedro_De_Uraba	0	
1457	Ecuador-Tungurahua	1	
1124	Colombia-Sucre-Santiago_De_Tolu	49	
191	Colombia-Antioquia-Zaragoza	0	
584	Colombia-Cordoba-Ayapel	0	
240	Colombia-Bogota-Teusaquillo_La_Esmeralda	0	
604	Colombia-Cordoba-Sahagun	2	
904	Colombia-Narino-Pasto	0	
445	Colombia-Caqueta-Albania	0	
1188	Colombia-Valle_Del_Cauca-Cali	12	
457	Colombia-Caqueta-San_Vicente_Del_Caguan	1	
832	Colombia-Meta-Acacias	1	
1759	United_States_Virgin_Islands-Saint_Thomas	20	
1241	Dominican_Republic-Barahona	2	
91	Colombia-Antioquia-Canasgordas	2	
257	Colombia-Bolivar-Cordoba	5	
...	...	...	
75	Colombia-Antioquia-Anza	0	
76	Colombia-Antioquia-Apartado	2	
1041	Colombia-Santander-El_Playon	0	
231	Colombia-Bogota-Ciudad_Bolivar_San_Francisco	0	
12	Argentina-Mendoza	13	
1121	Colombia-Sucre-San_Marcos	33	
997	Colombia-Risaralda-Belen_De_Umbria	2	

461	Colombia-Caqueta-Valparaiso	0
1600	Nicaragua-Rivas	5
1698	United_States-Mississippi	0
1637	Panama-Metro-Calidonia	2
1078	Colombia-Santander-Rionegro	0
1682	United_States-Delaware	0
1659	Panama-Panama_Oeste-Vista_Alegre	9
1003	Colombia-Risaralda-Mistrato	2
850	Colombia-Meta-Puerto_Lleras	0
1601	Nicaragua-Rivas-Cardenas	1
1641	Panama-Metro-Gamboa	1
899	Colombia-Narino-Mallama	0
1134	Colombia-Tolima-Anzoategui	1
378	Colombia-Boyaca-Santa_Maria	0
949	Colombia-Norte_Santander-Lourdes	0
263	Colombia-Bolivar-Mahates	30
1579	Nicaragua-Chinandega-San_Pedro_del_Norte	1
2	Argentina-Catamarca	14
6	Argentina-Corrientes	20
494	Colombia-Cauca-Guachene	0
1534	Haiti-Ouest-Delmas	13
594	Colombia-Cordoba-Los_Cordobas	0
916	Colombia-Narino-San_Pablo	0

	cases_total	date	zika_bool	max_temp	max_temp1	max_temp2	\
609	421	2016-01-16	1	0.112614	0.112823	0.052550	
1192	685	2016-02-20	1	0.122832	0.044059	0.042724	
46	263516	2016-02-13	1	0.008017	0.008518	0.022547	
170	31	2016-03-05	1	0.110220	0.111384	0.064532	
781	99	2016-01-23	1	0.122610	0.121333	0.120056	
890	0	2016-01-09	0	0.097630	0.102837	0.104139	
337	0	2016-01-09	0	0.108539	0.109832	0.111124	
905	15	2016-03-19	1	0.032766	0.032766	0.031401	
135	5	2016-05-28	1	0.096923	0.099543	0.028815	
124	20	2016-02-13	1	0.040263	0.040263	0.114078	
937	570	2016-02-06	1	0.041884	0.120417	0.119108	
140	800	2016-01-23	1	0.113845	0.113845	0.116403	
896	0	2016-01-09	0	0.097973	0.103198	0.104504	
670	57	2016-02-13	1	0.017899	0.019604	0.062220	
162	483	2016-01-23	1	0.113670	0.113670	0.116224	
1457	1	2016-05-18	1	0.221692	0.215358	0.218525	
1124	265	2016-01-09	1	0.117034	0.120809	0.122068	
191	933	2016-02-06	1	0.109440	0.111159	0.053145	
584	990	2016-01-16	1	0.112578	0.112787	0.052533	
240	0	2016-01-09	0	0.010936	0.010779	0.011092	
604	394	2016-01-09	1	0.103220	0.106549	0.107659	
904	85	2016-02-13	1	0.028719	0.028719	0.089469	
445	59	2016-02-27	1	0.118396	0.115794	0.042935	

1188	123757	2016-01-09	1	0.017437	0.018230	0.018032
457	1347	2016-01-09	1	0.107176	0.107636	0.042206
832	6527	2016-01-09	1	0.114123	0.111559	0.115406
1759	1533	2016-02-16	1	0.001092	0.001092	0.001092
1241	552	2016-02-13	1	0.015021	0.015021	0.041672
91	50	2016-01-09	1	0.108848	0.110144	0.111439
257	499	2016-01-09	1	0.112147	0.112317	0.050106
...	...	...	...	...	...	...
75	344	2016-05-07	1	0.036821	0.036821	0.042276
76	6087	2016-01-09	1	0.106930	0.105755	0.105755
1041	1214	2016-01-30	1	0.104690	0.105983	0.104690
231	0	2016-01-09	0	0.010936	0.010779	0.011092
12	284	2016-03-19	1	0.011009	0.010223	0.011402
1121	1742	2016-01-09	1	0.110979	0.111319	0.045552
997	126	2016-01-09	1	0.108098	0.110642	0.111913
461	58	2016-02-13	1	0.030918	0.030918	0.099477
1600	5	2016-05-05	1	0.177721	0.177721	0.182799
1698	0	2016-03-30	0	0.002849	0.002692	0.002927
1637	2	2016-04-11	1	0.002103	0.002103	0.002103
1078	543	2016-02-06	1	0.022354	0.071923	0.072894
1682	0	2016-02-24	0	0.000429	0.000195	0.000039
1659	9	2016-04-11	1	0.009131	0.009131	0.009131
1003	39	2016-01-09	1	0.095766	0.094454	0.094454
850	120	2016-01-30	1	0.120959	0.118385	0.115811
1601	1	2016-03-14	1	0.291514	0.291514	0.283185
1641	1	2016-04-11	1	0.195730	0.195730	0.195730
899	0	2016-01-09	0	0.090577	0.082585	0.079921
1134	26	2016-01-09	1	0.111825	0.119537	0.118251
378	73	2016-02-27	1	0.114928	0.114928	0.040347
949	71	2016-02-06	1	0.041845	0.120304	0.118996
263	862	2016-01-09	1	0.113374	0.113374	0.115922
1579	1	2016-02-12	1	0.275263	0.283604	0.275263
2	212	2016-03-19	1	0.025989	0.025151	0.025989
6	285	2016-03-19	1	0.022448	0.018828	0.020276
494	0	2016-01-09	0	0.112328	0.117433	0.116157
1534	13	2016-02-03	1	0.001327	0.001327	0.001327
594	56	2016-02-06	1	0.045377	0.120574	0.121871
916	9	2016-04-30	1	0.043562	0.044924	0.043562

	mean_temp	mean_temp1	...	precipitation2	wind	wind1	\
609	0.107448	0.106978	...	0.000019	0.010437	0.010646	
1192	0.106810	0.036048	...	0.000000	0.008011	0.013351	
46	0.007015	0.007265	...	0.000030	0.002505	0.002505	
170	0.100234	0.101883	...	0.016547	0.007224	0.008389	
781	0.111116	0.109838	...	0.000920	0.003832	0.003832	
890	0.087216	0.092423	...	0.000000	0.003905	0.014319	
337	0.096910	0.098202	...	0.000000	0.005169	0.009045	
905	0.027305	0.027305	...	0.000000	0.006826	0.006826	

135	0.085135	0.085135	...	0.034264	0.005239	0.005239
124	0.032210	0.032210	...	0.000000	0.012079	0.013421
937	0.036649	0.108637	...	0.000000	0.010471	0.007853
140	0.104891	0.104891	...	0.000000	0.011512	0.008954
896	0.087522	0.092747	...	0.000000	0.003919	0.014369
670	0.012785	0.011933	...	0.000170	0.009376	0.006819
162	0.104729	0.104729	...	0.000000	0.011495	0.008940
1457	0.190022	0.193189	...	0.000127	0.012668	0.006334
1124	0.106967	0.106967	...	0.000000	0.002517	0.001258
191	0.099241	0.101262	...	0.020284	0.007980	0.009700
584	0.107414	0.106944	...	0.000019	0.010434	0.010642
240	0.009217	0.009061	...	0.000000	0.000937	0.001094
604	0.094341	0.094341	...	0.000000	0.002220	0.001110
904	0.023196	0.022091	...	0.000000	0.007732	0.007732
445	0.106687	0.107988	...	0.000000	0.014312	0.014312
1188	0.015456	0.015852	...	0.000000	0.000991	0.000991
457	0.102093	0.103919	...	0.000286	0.013596	0.015251
832	0.103865	0.101301	...	0.000000	0.001282	0.001282
1759	0.000936	0.001014	...	0.000456	0.000741	0.000624
1241	0.012598	0.012114	...	0.000000	0.005815	0.006784
91	0.097186	0.098481	...	0.000000	0.005183	0.009071
257	0.107233	0.106609	...	0.000015	0.010401	0.010570
...	...	...	...	...	...	...
75	0.030002	0.030002	...	0.000000	0.006819	0.008182
76	0.098705	0.098705	...	0.000000	0.008225	0.009400
1041	0.094351	0.095643	...	0.000013	0.006462	0.006462
231	0.009217	0.009061	...	0.000000	0.000937	0.001094
12	0.008650	0.008257	...	0.012782	0.002359	0.001966
1121	0.101322	0.101662	...	0.000136	0.008807	0.007670
997	0.095381	0.096653	...	0.000051	0.007630	0.007630
461	0.024197	0.024197	...	0.000081	0.006721	0.006721
1600	0.152332	0.157410	...	0.000000	0.050777	0.050777
1698	0.002419	0.002146	...	0.000063	0.000312	0.000312
1637	0.001793	0.001793	...	0.000000	0.000618	0.000742
1078	0.017495	0.064147	...	0.000000	0.004860	0.003888
1682	0.000273	0.000039	...	0.000396	0.000702	0.000624
1659	0.007788	0.007788	...	0.000000	0.002686	0.003223
1003	0.087895	0.089207	...	0.000000	0.007871	0.006559
850	0.106804	0.104230	...	0.000000	0.002574	0.001287
1601	0.241540	0.241540	...	0.000000	0.141592	0.141592
1641	0.166946	0.166946	...	0.000000	0.057568	0.069081
899	0.073261	0.071929	...	0.000040	0.003996	0.006660
1134	0.098971	0.105398	...	0.000000	0.006427	0.007712
378	0.105147	0.105147	...	0.000000	0.007336	0.008558
949	0.036614	0.108535	...	0.000000	0.010461	0.007846
263	0.105731	0.105731	...	0.000000	0.008917	0.007643
1579	0.225215	0.233556	...	0.000000	0.150143	0.125120
2	0.020959	0.019282	...	0.013414	0.019282	0.016767

```
6      0.018828    0.015207    ...           0.024643  0.007241  0.007241
494    0.099563    0.102116    ...           0.000000  0.006382  0.006382
1534   0.001161    0.001161    ...           0.000000  0.000580  0.000580
594    0.037598    0.107609    ...           0.000000  0.005186  0.002593
916    0.038117    0.039478    ...           0.000000  0.004084  0.006807

         wind2  density_per_km  airport_dist_any  airport_dist_large  \
609   0.012837        0.042008      1.914334e-04        1.493631e-02
1192  0.010681        0.060120      1.451688e-04        1.009356e-02
46    0.001253        0.377058      2.702702e-08        3.665017e-06
170   0.012511        0.039093      3.647949e-04        7.797043e-03
781   0.002554        0.067342      5.473314e-04        9.464741e-03
890   0.013017        0.113540      1.877327e-04        6.398338e-03
337   0.006461        0.112776      2.826088e-04        4.960211e-03
905   0.004096        0.065199      1.132659e-04        5.319330e-03
135   0.006549        0.180592      7.588692e-05        4.511838e-03
124   0.006710        0.020645      1.258380e-03        1.170411e-02
937   0.007853        0.027094      2.863213e-04        1.301118e-02
140   0.008954        0.067835      4.906537e-04        9.146215e-03
896   0.013063        0.077362      7.349187e-05        3.769955e-03
670   0.004262        0.777745      1.228168e-05        1.228168e-05
162   0.008940        0.087114      4.192098e-04        1.232731e-02
1457  0.009501        0.361223      2.648251e-04        1.097393e-03
1124  0.001258        0.167543      3.529400e-05        1.845051e-02
191   0.014753        0.042150      2.008246e-05        1.086037e-02
584   0.012833        0.047811      1.562543e-04        1.825756e-02
240   0.001250        0.993049      9.962851e-07        9.962851e-07
604   0.001110        0.493904      1.780680e-04        1.721931e-02
904   0.005523        0.572676      3.995342e-05        3.238075e-03
445   0.024720        0.085672      2.032177e-05        5.741895e-02
1188  0.000991        0.988164      6.111224e-06        1.436909e-03
457   0.023356        0.008998      2.000685e-06        9.315665e-03
832   0.002565        0.105827      6.202945e-05        8.330259e-04
1759  0.000468        0.032568      2.428701e-07        4.826915e-05
1241  0.004846        0.993463      8.839264e-07        1.018604e-03
91    0.006479        0.083805      6.101362e-04        1.001375e-02
257   0.012832        0.095753      1.924451e-04        1.721455e-02
...        ...             ...               ...                 ...
75    0.010910        0.060491      1.027649e-04        7.457475e-03
76    0.008225        0.396652      1.568026e-05        1.059921e-02
1041  0.006462        0.104203      1.529288e-04        1.105753e-02
231   0.001250        0.993049      2.937905e-06        2.937905e-06
12    0.002359        0.901357      2.395797e-06        4.326036e-02
1121  0.010114        0.106629      6.107909e-04        2.175702e-02
997   0.008902        0.194437      2.126065e-04        4.088768e-03
461   0.005377        0.093074      4.614155e-04        4.058585e-03
1600  0.081244        0.830930      3.151978e-03        7.355376e-02
1698  0.000312        0.000533      1.623174e-05        1.964275e-05
```

1637	0.000680	0.999960	2.727602e-08	2.042180e-06
1078	0.002916	0.684243	2.626528e-06	3.489189e-03
1682	0.000663	0.002860	2.022796e-06	3.030402e-06
1659	0.002954	0.999243	5.996059e-06	3.217189e-05
1003	0.006559	0.038873	3.232409e-04	4.417270e-03
850	0.002574	0.005450	8.791065e-04	3.407629e-03
1601	0.158250	0.442893	1.100872e-02	1.465900e-01
1641	0.063324	0.551818	2.321460e-04	5.777337e-04
899	0.007992	0.018362	1.529841e-04	2.472288e-03
1134	0.007712	0.089626	5.879469e-05	1.158589e-03
378	0.012226	0.345134	1.819820e-05	3.845869e-02
949	0.007846	0.050169	1.341435e-04	1.601297e-02
263	0.007643	0.105551	1.863950e-04	2.410846e-02
1579	0.116778	0.535184	1.493830e-02	3.980231e-02
2	0.015929	0.385776	1.345324e-05	7.781874e-02
6	0.005793	0.606637	3.801470e-06	3.921266e-02
494	0.006382	0.153428	2.148367e-04	9.586271e-03
1534	0.000580	0.999992	6.184469e-08	2.886912e-04
594	0.002593	0.119000	3.688018e-04	1.193066e-02
916	0.006807	0.035721	2.991202e-04	1.055737e-02

	mosquito_dist	gdp	gdp_ppp
609	1.263290e-03	0.380295	0.870047
1192	9.702158e-05	0.389189	0.890395
46	3.236365e-06	0.450859	0.807774
170	3.601614e-04	0.381489	0.872778
781	1.151520e-03	0.372301	0.851759
890	1.140826e-03	0.379457	0.868129
337	2.932230e-04	0.376658	0.861726
905	1.731369e-03	0.397970	0.910485
135	1.007610e-04	0.381800	0.873490
124	5.816926e-04	0.391220	0.895042
937	3.341095e-04	0.381538	0.872891
140	1.411797e-04	0.372873	0.853068
896	1.164197e-03	0.380787	0.871173
670	4.410307e-05	0.248454	0.568418
162	1.961897e-04	0.372301	0.851757
1457	1.764580e-02	0.314487	0.585268
1124	7.363548e-04	0.366832	0.839247
191	6.278564e-04	0.382605	0.875333
584	2.703757e-03	0.380174	0.869769
240	2.253716e-05	0.045539	0.104185
604	1.997081e-03	0.323533	0.740185
904	5.081525e-04	0.321979	0.736630
445	1.819712e-04	0.379259	0.867676
1188	2.900185e-08	0.057761	0.132148
457	1.218065e-03	0.382914	0.876038
832	2.225923e-07	0.373786	0.855156

```
1759   4.348869e-05  0.706727  0.706727
1241   5.655294e-04  0.033047  0.072635
91     4.999855e-04  0.377728  0.864174
257    1.740576e-03  0.378982  0.867043
...             ...       ...       ...
75     7.650188e-05  0.397527  0.909470
76     6.848525e-05  0.342529  0.783645
1041   1.202695e-04  0.376756  0.861951
231    1.706072e-05  0.045539  0.104185
12     2.974757e-03  0.252617  0.347530
1121   3.660728e-03  0.380906  0.871444
997    6.046823e-04  0.370713  0.848126
461    4.958940e-04  0.391858  0.896501
1600   1.433686e-03  0.064487  0.162995
1698   1.538185e-03  0.707084  0.707084
1637   5.335609e-04  0.003358  0.005634
1078   4.241617e-05  0.283317  0.648178
1682   5.048434e-05  0.707102  0.707102
1659   2.552579e-03  0.014583  0.024466
1003   8.148085e-04  0.382408  0.874882
850    8.404499e-04  0.375100  0.858162
1601   4.757075e-03  0.105778  0.267360
1641   5.660856e-02  0.312592  0.524441
899    1.612916e-03  0.388282  0.888321
1134   2.454585e-04  0.374677  0.857194
378    4.806488e-04  0.356400  0.815379
949    2.015855e-04  0.381181  0.872073
263    1.408372e-04  0.371333  0.849543
1579   1.245044e-02  0.105935  0.267756
2      9.966055e-04  0.538649  0.741030
6      2.412000e-03  0.465263  0.640072
494    1.486684e-04  0.372085  0.851264
1534   3.287891e-08  0.000361  0.000780
594    1.002304e-03  0.377929  0.864635
916    2.531160e-04  0.396825  0.907866

[364 rows x 29 columns]
```

In [88]:
```
data_names = ['unbalanced','upsampled_3','upsampled_5','upsampled_7',
              'dwnsampled_3', 'dwnsampled_5', 'dwnsampled_7']
xdata_list = [train[feat_cols], X_resampled_3, X_resampled_5, X_resampled_7,
              X_dwnresampled_3, X_dwnresampled_5, X_dwnresampled_7]
ydata_list = [train['zika_bool'], Y_resampled_3, Y_resampled_5, Y_resampled_7,
              Y_dwnresampled_3, Y_dwnresampled_5, Y_dwnresampled_7]
```

In [89]:
```
def error_calc(model, label, Ytest=Ytest):

    Ypred = model.predict(test[feat_cols])
```

```
            cm = confusion_matrix(Ytest, Ypred)
            f1 = f1_score(Ytest, Ypred)
            accuracy = accuracy_score(Ytest, Ypred)
            precision = precision_score(Ytest, Ypred)
            recall = recall_score(Ytest, Ypred)

            Yplot = model.predict_proba(Xtest[feat_cols])[:,1]

            xdata, ydata, _ = roc_curve(Ytest, Yplot)
            auc = roc_auc_score(Ytest, Yplot)

            df = pd.Series({'data':label,
                            'cm': cm, 'f1': f1,
                            'accuracy':accuracy, 'auc':auc,
                            'precision':precision, 'recall':recall,
                            'roc_x': xdata, 'roc_y': ydata, 'model':model})
            return df
```

```
In [91]: xdata_list[1]
```

```
Out[91]: array([[1.09356359e-02, 1.07794126e-02, 1.10918593e-02, ...,
                 1.61505109e-05, 4.55391125e-02, 1.04185366e-01],
                [1.22975877e-01, 1.22975877e-01, 1.20440292e-01, ...,
                 1.72438681e-04, 3.69561527e-01, 8.45490849e-01],
                [2.43025004e-02, 2.53826115e-02, 9.18094460e-03, ...,
                 2.96841767e-04, 4.85969000e-01, 8.70677581e-01],
                ...,
                [9.18601117e-02, 9.05478244e-02, 9.31723991e-02, ...,
                 1.24894395e-04, 3.82531751e-01, 8.75164407e-01],
                [9.17498121e-02, 9.04391005e-02, 9.30605237e-02, ...,
                 5.09256924e-04, 3.82072432e-01, 8.74113567e-01],
                [9.17572986e-02, 9.04464801e-02, 9.30681172e-02, ...,
                 4.07556104e-04, 3.82103608e-01, 8.74184892e-01]])
```

1.2 Logistic regression

```
In [92]: logistic_list = list()
```

```
        param_grid = [{'C':[0.001,0.01,0.1,1,10,100,1000], 'penalty':['l1'], 'solver':['libli
                      {'C':[0.001,0.01,0.1,1,10,100,1000], 'penalty':['l2']}]

        MOD = LogisticRegression()
        CV = GridSearchCV(MOD, param_grid)

        for Xtrain,Ytrain,name in zip(xdata_list, ydata_list, data_names):
            #print (name,Xtrain,Ytrain,)
```

```
            CV.fit(Xtrain, Ytrain)

            df = error_calc(CV, name)

            logistic_list.append(df)
```

In [57]:
```
#CV.fit(xdata_list[0],ydata_list[0])
```

Out[57]:
```
GridSearchCV(cv=None, error_score='raise',
       estimator=LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercep
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
          penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
          verbose=0, warm_start=False),
       fit_params={}, iid=True, n_jobs=1,
       param_grid=[{'C': [0.001, 0.01, 0.1, 1, 10, 100, 1000], 'penalty': ['l1'], 'sol
       pre_dispatch='2*n_jobs', refit=True, scoring=None, verbose=0)
```

In [59]:
```
logistic_results = (pd.concat(logistic_list, axis=1).T
                        .set_index('data'))
logistic_results[['accuracy','f1','recall','precision','auc', 'cm']]
```

Out[59]:

data	accuracy	f1	recall	precision	auc	cm
unbalanced	0.815934	0.898638	1	0.815934	0.5	[[0, 67], [0, 297]]
upsampled_3	0.651099	0.751468	0.646465	0.897196	0.782853	[[45, 22], [105, 192]]
upsampled_5	0.656593	0.757282	0.656566	0.894495	0.774562	[[44, 23], [102, 195]]
upsampled_7	0.648352	0.75	0.646465	0.893023	0.769436	[[44, 23], [105, 192]]
dwnsampled_3	0.851648	0.912621	0.949495	0.878505	0.769184	[[28, 39], [15, 282]]
dwnsampled_5	0.818681	0.890728	0.905724	0.876221	0.760842	[[29, 38], [28, 269]]
dwnsampled_7	0.785714	0.866894	0.855219	0.878893	0.75848	[[32, 35], [43, 254]]

In [93]:
```
tmp = logistic_results.loc['unbalanced','model']
```

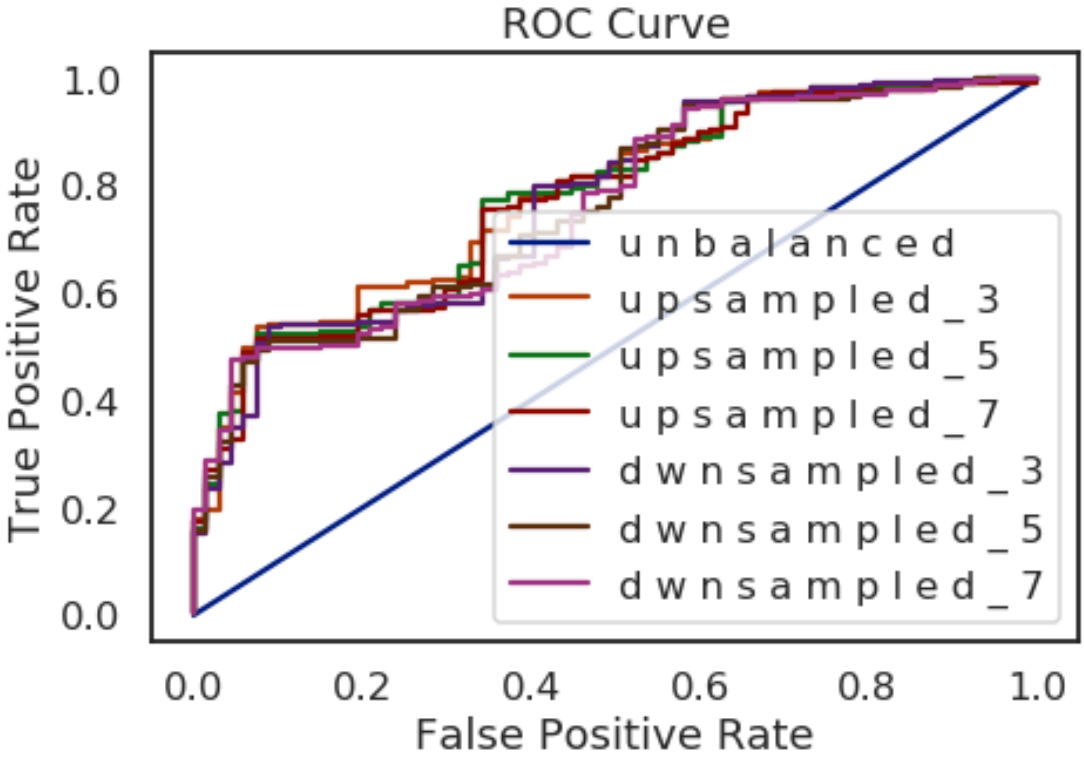
In [94]:
```
a = tmp.best_estimator_
```

In [95]:
```
tmp2 = np.array(feat_cols)
tmp2[np.abs(a.coef_).argsort()]
```

Out[95]:
```
array([['dew_point2', 'dew_point1', 'mean_temp1', 'mean_temp2',
        'density_per_km', 'gdp', 'min_temp', 'gdp_ppp',
```

```
       'airport_dist_large', 'max_temp', 'max_temp1', 'min_temp1',
       'wind1', 'mean_temp', 'airport_dist_any', 'dew_point',
       'precipitation2', 'mosquito_dist', 'wind2', 'max_temp2',
       'min_temp2', 'precipitation', 'wind', 'precipitation1']],
      dtype='<U18')
```

In [96]:
```
f = plt.figure()
f.set_size_inches(7,5)
ax = plt.axes()

colors = sns.color_palette()

for idx,dat in logistic_results.iterrows():
    label = ' '.join(idx)
    xdata = dat.roc_x
    ydata = dat.roc_y
    ax.plot(xdata, ydata, label=label)

ax.legend(loc=0)

_ = ax.set(xlabel='False Positive Rate',
       ylabel='True Positive Rate',
       title='ROC Curve')

plt.tight_layout()
```



1.3 Random Forest Classifier

In [66]:
```
random_forest_list = list()

param_grid = {'max_depth':list(range(5,10)), 'n_estimators':[300,400,500],
              'class_weight':['balanced','balanced_subsample']}

MOD = RandomForestClassifier(n_jobs=-1,random_state=42, oob_score=True)
CV = GridSearchCV(MOD, param_grid)

for Xtrain,Ytrain,name in zip(xdata_list, ydata_list, data_names):
    print (name)
    CV.fit(Xtrain, Ytrain)

    df = error_calc(CV, name)

    random_forest_list.append(df)
```

```
unbalanced
upsampled_3
upsampled_5
upsampled_7
dwnsampled_3
dwnsampled_5
dwnsampled_7
```

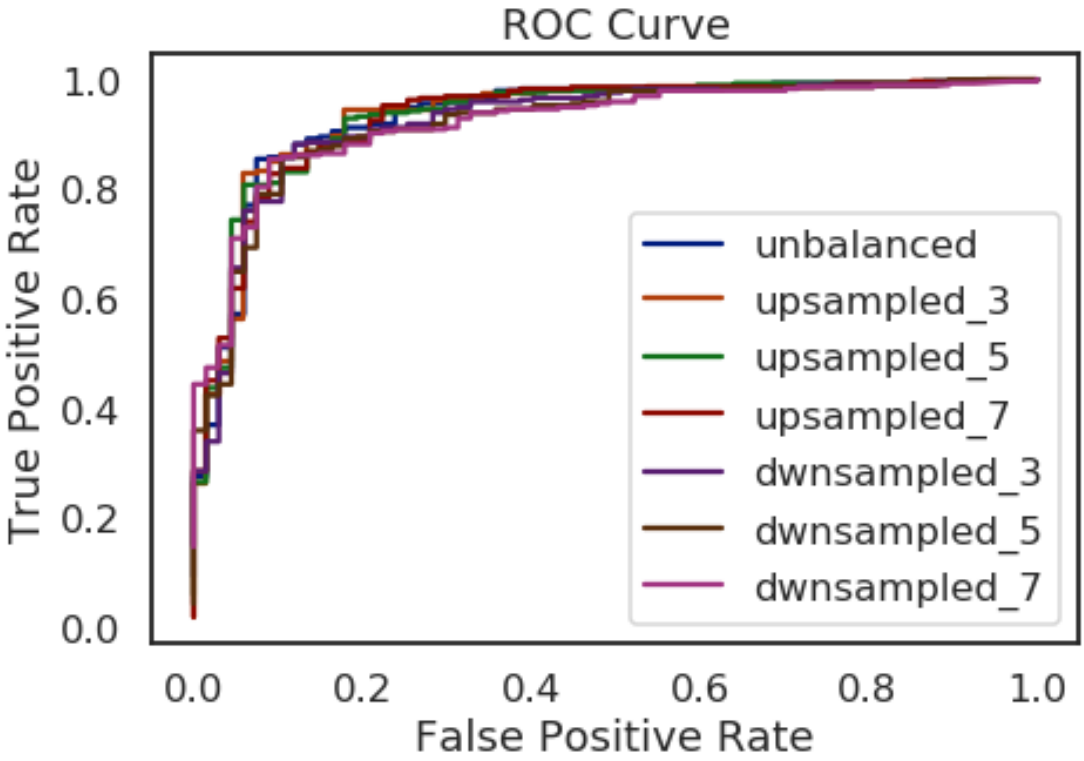
In [68]:
```
random_forest_results = (pd.concat(random_forest_list, axis=1).T
                           .set_index('data'))
random_forest_results[['accuracy','f1','recall','precision','auc', 'cm']]
```

Out[68]:

data	accuracy	f1	recall	precision	auc	cm
unbalanced	0.909341	0.944351	0.942761	0.945946	0.933162	[[51, 16], [17, 280]]
upsampled_3	0.901099	0.938144	0.919192	0.957895	0.934419	[[55, 12], [24, 273]]
upsampled_5	0.881868	0.924956	0.892256	0.960145	0.93364	[[56, 11], [32, 265]]
upsampled_7	0.870879	0.918261	0.888889	0.94964	0.932384	
dwnsampled_3	0.881868	0.924162	0.882155	0.97037	0.923715	
dwnsampled_5	0.840659	0.894545	0.828283	0.972332	0.920071	
dwnsampled_7	0.821429	0.879406	0.79798	0.979339	0.922031	

```
upsampled_7   [[53, 14], [33, 264]]
dwnsampled_3   [[59, 8], [35, 262]]
dwnsampled_5   [[60, 7], [51, 246]]
dwnsampled_7   [[62, 5], [60, 237]]
```

In [69]:
```
f = plt.figure()
f.set_size_inches(7,5)
ax = plt.axes()

colors = sns.color_palette()

for idx,dat in random_forest_results.iterrows():
    label = idx
    xdata = dat.roc_x
    ydata = dat.roc_y
    ax.plot(xdata, ydata, label=label)

ax.legend(loc=0)

_ = ax.set(xlabel='False Positive Rate',
       ylabel='True Positive Rate',
       title='ROC Curve')

plt.tight_layout()
```



1.4 AdaBoost classification

In [74]:
```
ada_list = list()

param_grid = {'learning_rate':[0.1, 0.3, 0.5, 0.7, 0.9, 1.0],
              'n_estimators':[300,400,500]}

# param_grid = {'learning_rate':[0.1],
#               'n_estimators':[500]}

tree = DecisionTreeClassifier(criterion='entropy', max_depth=1)
MOD = AdaBoostClassifier(base_estimator=tree,
                         random_state=42)
CV = GridSearchCV(MOD, param_grid)

for Xtrain,Ytrain,name in zip(xdata_list, ydata_list, data_names):
    print (name)
    if name == 'unbalanced':

        #Xtrain = pd.DataFrame(Xtrain,columns=feat_cols)
        Xtrain.to_pickle('./pkl/13_model_tuning_Xtrain_ada_inner_{}.pkl'.format(name))

        #Ytrain = pd.DataFrame(Ytrain,columns=['zika_bool'])
        Ytrain.to_pickle('./pkl/13_model_tuning_Ytrain_ada_inner_{}.pkl'.format(name))

        Xtest.to_pickle('./pkl/13_model_tuning_Xtest_ada_inner_{}.pkl'.format(name))
        Ytest.to_pickle('./pkl/13_model_tuning_Ytest_ada_inner_{}.pkl'.format(name))

        CV.fit(Xtrain[feat_cols], Ytrain)

        with open('./pkl/13_model_tuning_ada_models_inner_{}.pkl'.format(name),'wb') a
            dill.dump(CV,fh)

        Ypred_test = pd.DataFrame(CV.predict(Xtest[feat_cols]),
                                  columns=['zika_bool'],
                                  index=Xtest.index)

        Ypred_train = pd.DataFrame(CV.predict(Xtrain[feat_cols]),
                                   columns=['zika_bool'],
                                   index=Xtrain.index)

        Ypred_test.to_pickle('./pkl/13_model_tuning_Ypred_test_ada_inner_{}.pkl'.forma
        Ypred_train.to_pickle('./pkl/13_model_tuning_Ypred_train_ada_inner_{}.pkl'.fo

        print (confusion_matrix(Ytest, Ypred_test))
```

```
                df = error_calc(CV, name)

                ada_list.append(df)

unbalanced
[[ 45  22]
 [  8 289]]
upsampled_3
```

```
        ---------------------------------------------------------------------------

        AttributeError                            Traceback (most recent call last)

        <ipython-input-74-b2227ea8dde2> in <module>()
         17
         18         #Xtrain = pd.DataFrame(Xtrain,columns=feat_cols)
    ---> 19         Xtrain.to_pickle('../pkl/13_model_tuning_Xtrain_ada_inner_{}.pkl'.format(na
         20
         21         #Ytrain = pd.DataFrame(Ytrain,columns=['zika_bool'])


        AttributeError: 'numpy.ndarray' object has no attribute 'to_pickle'
```

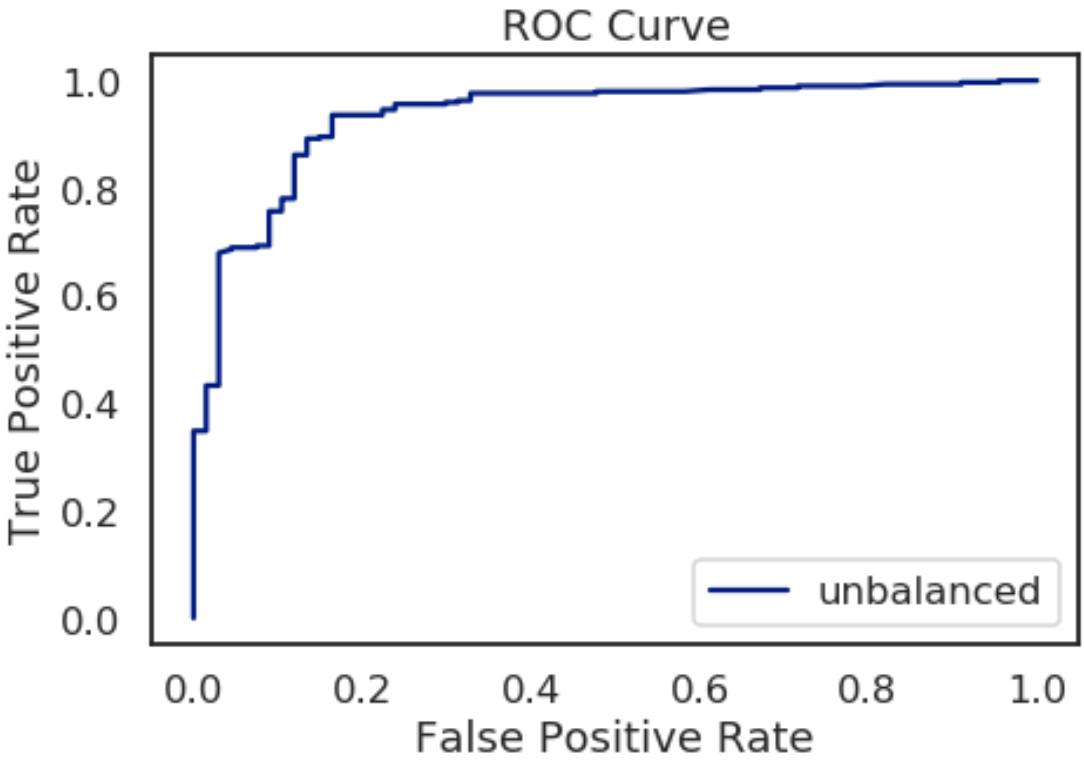
```
In [75]: ada_results = pd.concat(ada_list, axis=1).T.set_index('data')
         ada_results[['accuracy','f1','recall','precision','auc', 'cm']]
```

Out[75]:

data	accuracy	f1	recall	precision	auc	cm
unbalanced	0.917582	0.950658	0.973064	0.92926	0.933263	[[45, 22], [8, 289]]

```
In [76]: f = plt.figure()
         f.set_size_inches(7,5)
         ax = plt.axes()

         colors = sns.color_palette()

         for idx,dat in ada_results.iterrows():
             label = idx
             xdata = dat.roc_x
             ydata = dat.roc_y
             ax.plot(xdata, ydata, label=label)
```

```
ax.legend(loc=0)

_ = ax.set(xlabel='False Positive Rate',
       ylabel='True Positive Rate',
       title='ROC Curve')

plt.tight_layout()
```



In [80]:
```
logistic_results.to_pickle('./pkl/13_model_tuning_logistic_models_df.pkl')

for idx,row in logistic_results.iterrows():
    print (idx)
    with open('./pkl/13_model_tuning_logistic_{}.pkl'.format(idx),'wb') as fh:
        dill.dump(row.model,fh)
```

```
unbalanced
upsampled_3
upsampled_5
upsampled_7
dwnsampled_3
dwnsampled_5
dwnsampled_7
```

```
In [79]: random_forest_results.to_pickle('./pkl/13_model_tuning_random_forest_models_df.pkl')

         for idx,row in random_forest_results.iterrows():
             print (idx)
             with open('./pkl/13_model_tuning_random_forest_{}.pkl'.format(idx),'wb') as fh:
                 dill.dump(row.model,fh)

unbalanced
upsampled_3
upsampled_5
upsampled_7
dwnsampled_3
dwnsampled_5
dwnsampled_7


In [78]: ada_results.to_pickle('./pkl/13_model_tuning_ada_models_df.pkl')

         for idx,row in ada_results.iterrows():
             print (idx)
             with open('./pkl/13_model_tuning_ada_models_{}.pkl'.format(idx),'wb') as fh:
                 dill.dump(row.model,fh)

unbalanced


In [ ]:
```