Metrics recorded throughout development of classifier

Removal of outliers' impact: F1 recorded

	Before	After
Gaussian NB	0.26	0.32
Bernoulli NB	0.37	0.37
RFC	0.19	0.23

Initial experimentation with PCA: F1 recorded

	No PCA	9 PCs	8 PCs	**7 PCs**	**6 PCs**	**5 PCs**	**4 PCs**	3 PCs	2 PCs
Gaussian NB	0.26	0.32	0.32	**0.32**	**0.32**	**0.32**	**0.32**	0.32	0.32
Bernoulli NB	0.37	0.37	0.37	**0.37**	**0.37**	**0.37**	**0.37**	0.37	0.37
RFC	0.19	0.22	0.22	**0.24**	**0.24**	**0.24**	**0.24**	0.23	0.24

The optimum region for F1 seems to be 4 to 7 components. Will use 7 to allow for more latitude tuning parameter max_features.

Random Forest tuning initial starter estimator: out of the box. These are the default parameters:

bootstrap: True,
class_weight: None,
criterion: 'gini',
max_depth: None,
max_features: 'auto',
max_leaf_nodes: None,
min_samples_leaf: 1,
min_samples_split: 2,
min_weight_fraction_leaf: 0.0,
n_estimators: 10,
n_jobs: 1,
oob_score: False,
random_state: None,
verbose: 0,
warm_start: False

RFC ()	Precision	Recall	F1	Accuracy
	0.41	0.14	0.21	0.86

Grid Search with grid {'criterion': ['gini, 'entropy']}
Best estimator returns 'entropy'

RFC ()	Precision	Recall	F1	Accuracy
	0.44	0.15	0.22	0.86

Criterion is hard-coded into starter estimator

Grid Search with grid {'class_weight': ['balanced', 'balanced_subsample', None]}
Best estimator returns 'balanced_subsample'

RFC (criterion = 'entropy')	Precision	Recall	F1	Accuracy
	0.42	0.11	0.18	0.86

Class weight is hard-coded into starter estimator despite performance depreciation. Will revisit at the end.

Grid Search with grid {'max_depth: [10, 20, 40, 45, 50]}
Best estimator returns 40

RFC (criterion = 'entropy', class_weight = 'balanced_subsample')	Precision	Recall	F1	Accuracy
	0.43	0.12	0.18	0.86

Max depth is hard-coded into starter estimator.

Grid Search with grid {'max_features: [4, 5, 6, 7, None]}
Best estimator returns 6

RFC (criterion = 'entropy', class_weight = 'balanced_subsample', max_depth = 40)	Precision	Recall	F1	Accuracy
	0.36	0.13	0.19	0.85

Max features is hard-coded into starter estimator despite performance depreciation. Will revisit at the end.

Grid Search with grid {' min_samples_split: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]}
Best estimator returns 2

RFC (criterion = 'entropy', class_weight = 'balanced_subsample', max_features = 6, max_depth = 40)	Precision	Recall	F1	Accuracy
	0.40	0.14	0.21	0.85

Grid Search with grid {‘ min_samples_leaf: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]}

Best estimator returns 4

RFC (criterion = ‘entropy’, class_weight = ‘balanced_subsample’, max_features = 6, max_depth = 40, min_samples_split = 2)	Precision	Recall	F1	Accuracy
	0.34	0.37	0.35	0.82

Min samples leaf is hard-coded into starter estimator

Grid Search with grid {‘ min_weight_fraction_leaf: [0.0, 0.1, 0.2, 0.3, 0.4, 0.5]}

Best estimator returns 0.1

RFC (criterion = ‘entropy’, class_weight = ‘balanced_subsample’, max_features = 6, max_depth = 40, min_samples_split = 2, min_samples_leaf = 4)	Precision	Recall	F1	Accuracy
	0.32	0.40	0.36	0.8

Min weight fraction leaf hard-coded into starter estimator

Grid Search with grid {n_estimators: [2, 4, 8, 9, 10]}

Best estimator returns 8

RFC (criterion = ‘entropy’, class_weight = ‘balanced_subsample’, max_features = 6, max_depth = 40, min_samples_split = 2, min_samples_leaf = 4, min_weight_fraction_leaf = 0.1)	Precision	Recall	F1	Accuracy
	0.32	0.44	0.37	0.81

N estimators hard-coded into starter estimator

Revisiting class weight.

Grid Search with grid {class_weight: [‘None’, ‘balanced’, ‘balanced_subsample’]}

Best estimator returns balanced.

RFC (criterion = ‘entropy’, max_features = 6, max_depth = 40, min_samples_split = 2, min_samples_leaf = 4, min_weight_fraction_leaf = 0.1, n_estimators = 8)	Precision	Recall	F1	Accuracy
	0.33	0.43	0.37	0.80

This is step is not a clear-cut choice. F1 is the same, and precision depreciated while recall improved. From to the SKLearn Documentation we have:

> "The "balanced" mode uses the values of y to automatically adjust weights inversely proportional to class frequencies in the input data as `n_samples / (n_classes * np.bincount(y))` The "balanced_subsample" mode is the same as "balanced" except that weights are computed based on the bootstrap sample for every tree grown."

The most sound approach seems to me to be considering the balance of classes overall, so I am hard-coding balanced as a parameter value for class weight.

Revisiting max features.
Grid Search with grid {max_features: [6, 7,None]}
Best estimator returns 7.

	Precision	Recall	F1	Accuracy
`RFC (criterion = 'entropy', max_features = 7, class_weight = 'balanced', max_depth = 40, min_samples_split = 2, min_samples_leaf = 4, min_weight_fraction_leaf = 0.1, n_estimators = 8)`	0.34	0.44	0.38	0.81

Grid Search with grid {max_leaf_nodes: [40, 43, 45, 50, 55, 60]}
Best estimator returns 45.

	Precision	Recall	F1	Accuracy
`RFC (criterion = 'entropy', class_weight = 'balanced', max_depth = 40, min_samples_split = 2, min_samples_leaf = 4, min_weight_fraction_leaf = 0.1, n_estimators = 8)`	0.32	0.43	0.37	0.81

Performance depreciated, will be leaving this parameter as default.

Tuned estimator:

```
RandomForestClassifier(
    criterion = 'entropy',
    class_weight = 'balanced',
    max_depth = 40,
    max_features = 7,
    min_samples_split = 2,
    min_weight_fraction_leaf=0.1,
    n_estimators = 8)
```