CAPSTONE 2: GENOTYPING SNP QC

CARSTEN BRUCKNER

JAN 2022

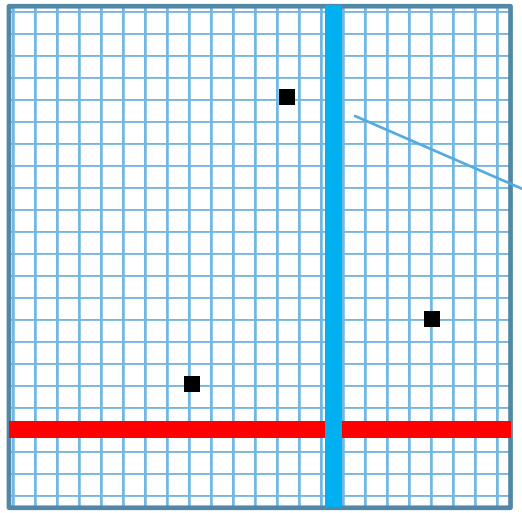
TYPICAL PRIMARY QUALITY CONTROL STEPS FOR GENETIC DATA

- instrument raw data
- remove outlier DNA samples
- remove likely unreliable probesets
- set to NoCall some individual calls

Can we identify unreliable probesets better?

1 - n DNA samples

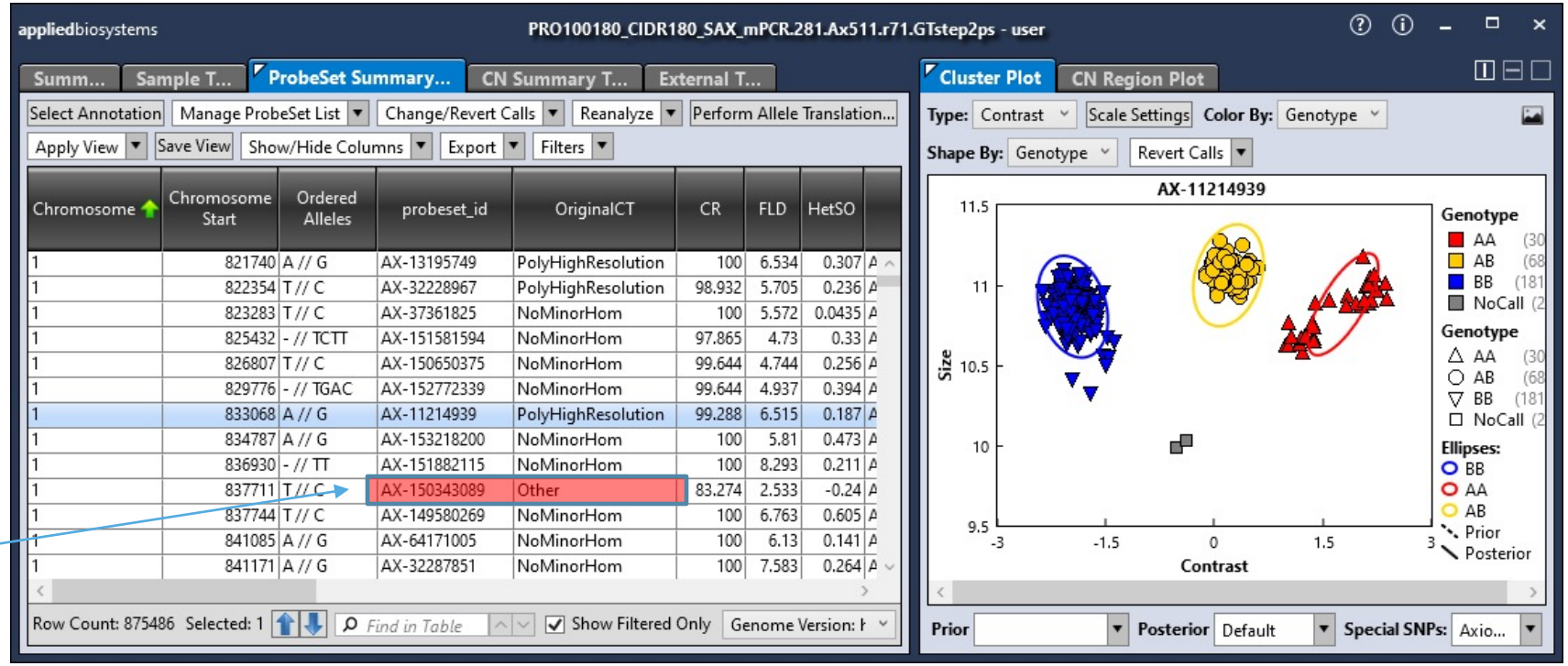
1 – m chromosome positions (each position measured by one or more “probesets”)



each square is one genotype call

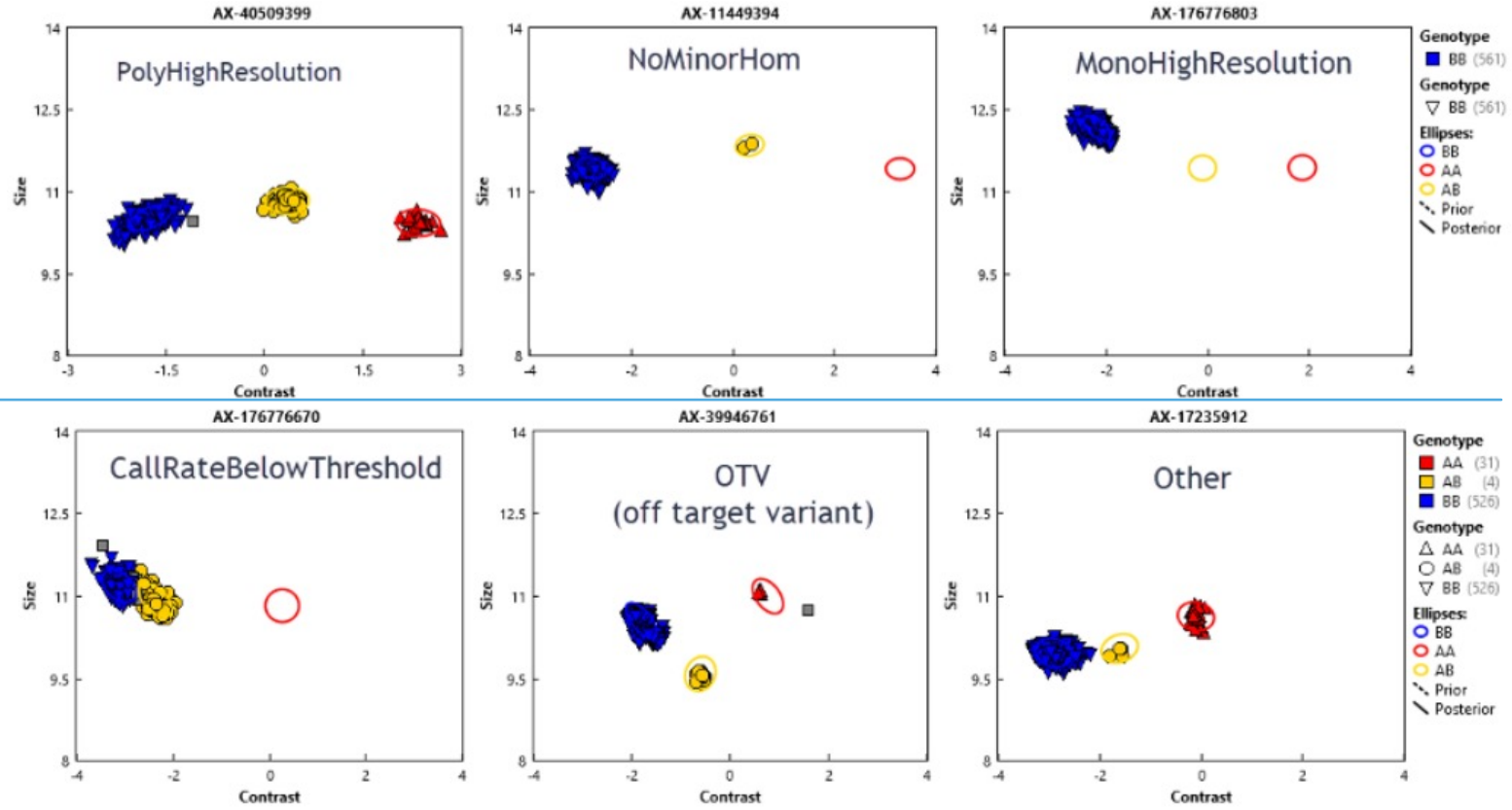
EXISTING HEURISTIC CLASSIFICATION PROBESETS BY DATA QUALITY

- each probeset_id measures many DNA samples (raw data points in cluster plot)
- Software tries to label each data point with a correct genotype (colors).
- If quality of cluster plot questionable, a non-recommended “OriginalCT” label is assigned to entire probeset_id



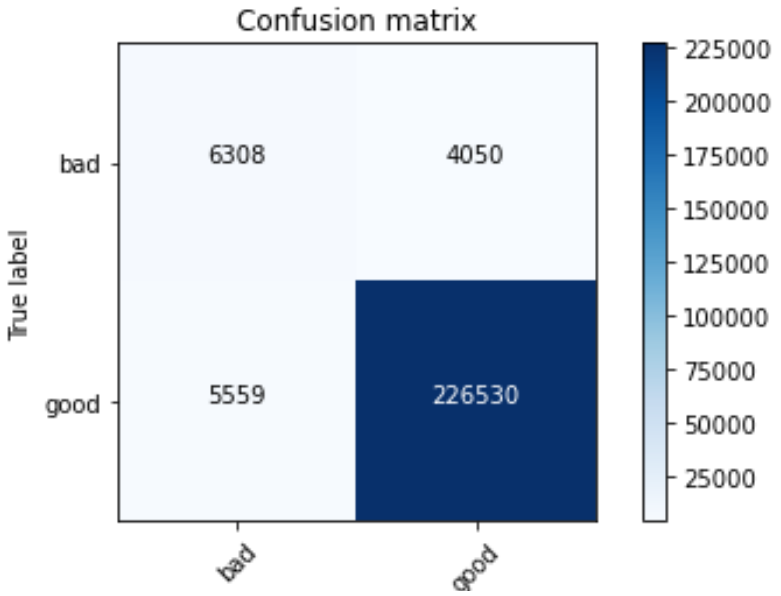
EXAMPLE GOOD AND BAD QUALITY BINS FOR PROBESETS

- Top row of plots are good quality bins
- Bottom row are bad quality



‘ORIGINALCT’ HEURISTIC CLASSIFICATION DOESN’T PERFECTLY SEPARATE GOOD FROM BAD PROBESETS

- for this project, “True label” a “good” probeset as having
  - good data completeness (Call Rate > 95% of samples)
  - good agreement with expected genotype (Concordance > 98.5%)
- expected genotype call for each data point is not available to the QC software
  - expected genotype is approximated by alternate technology measurement of same DNA
- A held-out test set of 242447 probesets from one dataset has a balanced classification accuracy of 79.3% (based on confusion matrix



Heuristic Classification
(derived from ‘OriginalCT’)

True	precision	recall	F-beta	support (n)
bad	0.531558	0.608998	0.567649	10358
good	0.982436	0.976048	0.979231	232089

balanced_accuracy: 79.3 %

OBJECTIVE: BUILD A MODEL THAT PREDICTS PROBESET QUALITY BETTER THAN EXISTING 'ORIGINALCT'

- The quality category to predict is "quality_binary_good", where observations have "1" category if [Call Rate (CR) > 95%] AND [Concordance to 1000Genomes reference data (CC) > 98.5%]. Otherwise quality_binary_good = "0".
- Will the model have a higher balanced_accuracy predicting "quality_binary_good" than using only one of the engineered features, namely 'OriginalCT.recommended_True' ?
  - balanced_accuracy is average of accuracy predicting "bad" probesets, and accuracy predicting "good" probesets

SOURCE DATA

- Thermo Fisher microarray Axiom_VA_MVPEF.r4_1
- Genotyping of coriell.org sample plates HAPMAPT01, T02, and T03
- Probeset summary statistics (raw features for model) available for 884158 probesets

- Independent genotypes from 1000Genomes Phase III project available for 851611 probesets
  - needed to evaluate concordance and generate predicted quality bin.

DATA CLEANING

- remove 10% of original observations (probeset rows) because they belong to at least one of these categories:
  - more than 2 alleles
  - on non-autosomal chromosomes (X,Y,mitochondrial)
  - non-diploid calls
  - no values for essential Concordance component of the predictor variable
  - unreliable values for essential Concordance component of the predictor variable

original probesets

Row Count per only_diploid_calls, OriginalCT

(None)

Data table: Ps.performance

only_diploid_calls	OriginalCT	(Row Count)
False	PolyHighResolution	117
	NoMinorHom	38
	MonoHighResolution	505
	CallRateBelowThres...	7
	Other	108
True	PolyHighResolution	270473
	NoMinorHom	508900
	MonoHighResolution	56373
	CallRateBelowThres...	2471
	AAvarianceX	889
	AAvarianceY	1464
	ABvarianceX	2529
	ABvarianceY	2889
	BBvarianceX	1436
	BBvarianceY	1450
	Other	32022
	OTV	2487
Grand total		884158

OriginalCT * only_diploid_calls

(Row Count)

trimmed probesets for evaluation

Row Count per only_diploid_calls, OriginalCT

(None)

Data table: inner_join

only_diploid_calls	OriginalCT	(Row Count)
True	PolyHighResolution	235576
	NoMinorHom	504202
	MonoHighResolution	28818
	CallRateBelowThreshold	2082
	Other	24931
	OTV	1954
	AAvarianceX	886
	AAvarianceY	1458
	ABvarianceX	2507
	ABvarianceY	2873
	BBvarianceX	1426
	BBvarianceY	1442
Grand total		808155

OriginalCT * only_diploid_calls

(Row Count)

FEATURE ENGINEERING

- create predictor quality_binary, where observations have “good” category if [Call Rate (CR) > 95%] AND [Concordance to 1000Genomes reference data (CC) > 98.5%].
- replace highly related count metrics n_AA, n_AB, n_BB with het_frac = n_AB/(n_AA + n_AB + n_BB).
- convert categorical columns to multiple Boolean (one-hot encoding)
- reset to Null in new columns any values that exist only due to Bayesian prior values and not due to the data being measured (for example, if there are no samples for AA cluster, then AA.meanX and AA.meanY metrics are not real values). New columns were created with .clean appended to name, original columns were removed.
- replace Null values with appropriate values:
  - use -999 for empty ['BB.meanX.clean','HomRO’]
  - use 0 for empty ['AB.meanX.abs_clean','AA.varX.clean’, 'AB.varX.clean’, 'BB.varX.clean', 'AA.varY.clean’, 'AB.varY.clean’, 'BB.varY.clean’]
  - use +999 for empty ['AA.meanX.clean','FLD','HetSO','MMD']

FEATURES USED FOR MODEL BUILDING

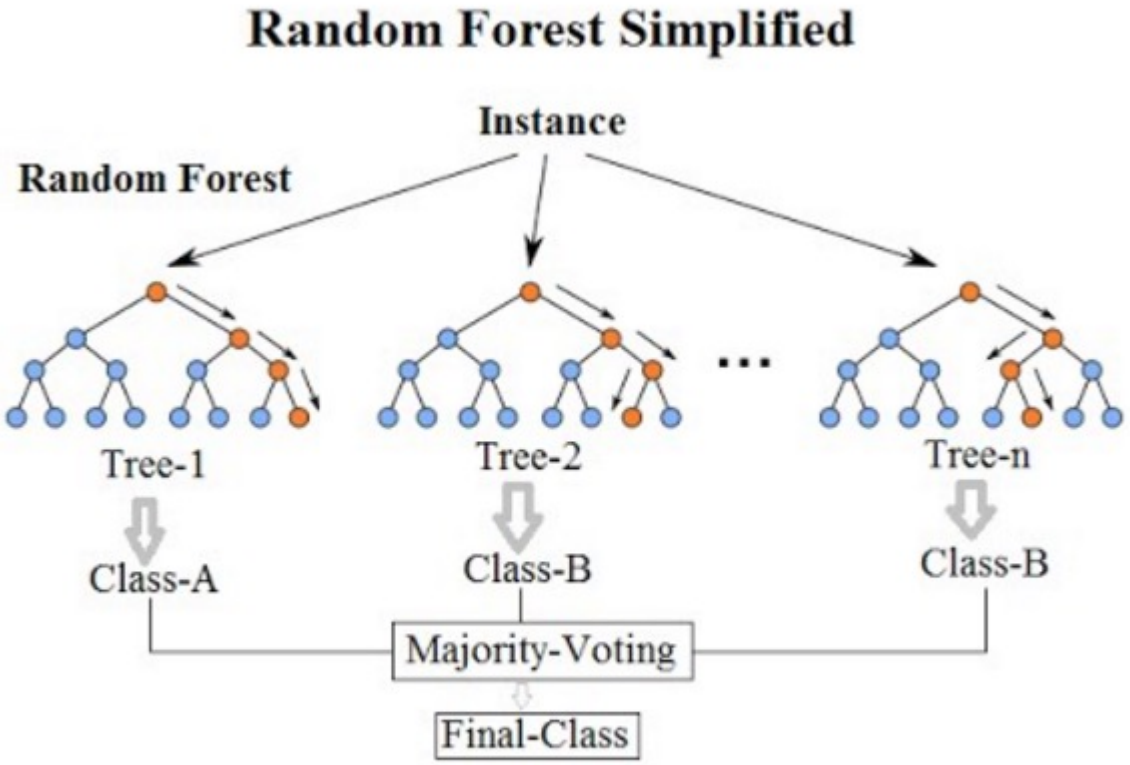
feature	type
CR	float
FLD	float
HetSO	float
MMD	float
het_frac	float
MinorAlleleFrequency	float
H.W.p-Value	float
AA.meanX.clean	float
AB.meanX.abs_clean	float
BB.meanX.clean	float
HomRO	float
AA.meanY.clean	float
AB.meanY.clean	float

feature	type
BB.meanY.clean	float
meanY	float
Hom.meanY.delta	float
AA.varX.clean	float
AB.varX.clean	float
BB.varX.clean	float
AA.varY.clean	float
AB.varY.clean	float
BB.varY.clean	float
OriginalCT.recommended_True	boolean
OriginalCT_AAvarianceX	boolean
OriginalCT_AAvarianceY	boolean
OriginalCT_ABvarianceX	boolean

feature	type
OriginalCT_ABvarianceY	boolean
OriginalCT_BBvarianceX	boolean
OriginalCT_BBvarianceY	boolean
OriginalCT_CallRateBelowThreshold	boolean
OriginalCT_MonoHighResolution	boolean
OriginalCT_NoMinorHom	boolean
OriginalCT_OTV	boolean
OriginalCT_Other	boolean
OriginalCT_PolyHighResolution	boolean
Nclus_1	boolean
Nclus_2	boolean
Nclus_3	boolean

RANDOM FOREST SUPERVISED LEARNING MODEL

- For each node of each decision tree,
  - select random subset of metrics
  - for each metric, find threshold that best separates good from bad probesets
  - use single metric+its threshold that best separates good from bad probesets
- train a forest of trees
- use majority voting to settle on final predicted quality of a probeset



https://en.wikipedia.org/wiki/Random_forest

SPLIT PROBESETS INTO TRAIN AND TEST SETS

- Split out 30% of the probesets just for model testing, preserving proportions of probesets in each OriginalCT
- Balance the good/bad probeset classes in training set by retaining 1 of every 15 good probesets

probeset count:	quality_binary	
OriginalCT	good	bad
NoMinorHom	498455	5747
PolyHighResolution	228775	6801
MonoHighResolution	27866	952
Other	8175	16756
ABvarianceY	2322	551
ABvarianceX	2251	256
AAvarianceY	1383	75
BBvarianceX	1359	67
BBvarianceY	1300	142
OTV	927	1027
AAvarianceX	812	74
CallRateBelowThreshold	0	2082
total	773625	34530
	95.7%	4.3%

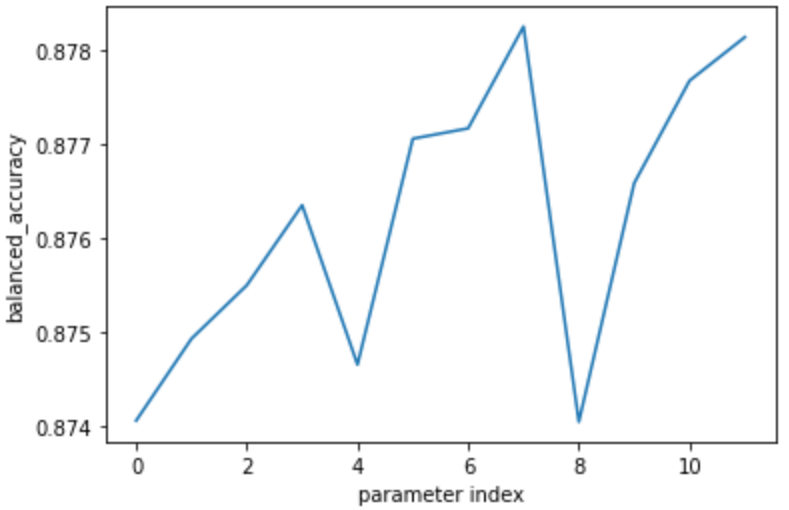
test (30% of original)	
quality_binary	
good	bad
149537	1724
68633	2040
8360	286
2453	5027
697	165
675	77
415	23
408	20
390	43
278	308
244	22
0	625
232090	10360
95.7%	4.3%

train_temp (70% of original)	
quality_binary	
good	bad
348919	4023
160143	4761
19506	666
5723	11729
1625	386
1576	179
968	53
951	47
910	99
649	719
568	52
0	1457
541538	24171
95.7%	4.3%

train (subsample good)	
quality_binary	
good	bad
36102	4023
	4761
	666
	11729
	386
	179
	53
	47
	99
	719
	52
	1457
36102	24171
59.9%	40.1%

RANDOM FOREST HYPERPARAMETER TUNING

parameter index	n_estimators (trees)	max_depth
1	25	15
1	50	15
2	100	15
3	800	15
4	25	25
5	50	25
6	100	25
7	800	25
8	25	None
9	50	None
10	100	None
11	800	None

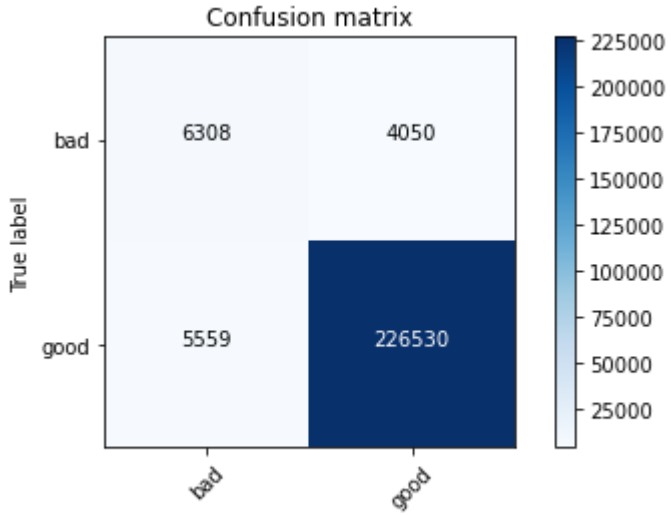


- 5-fold cross validation performed within train_subsample set for each set of evaluated parameters.
- other parameters were defaults
- more trees give better balanced accuracy, at the expense of prediction time
- selected n_estimators=800, max_depth=25 for eventual model (parameter index 7)

MODEL PERFORMANCE ON INDEPENDENT TEST SET

Relative to benchmark, this Random Forest model has:

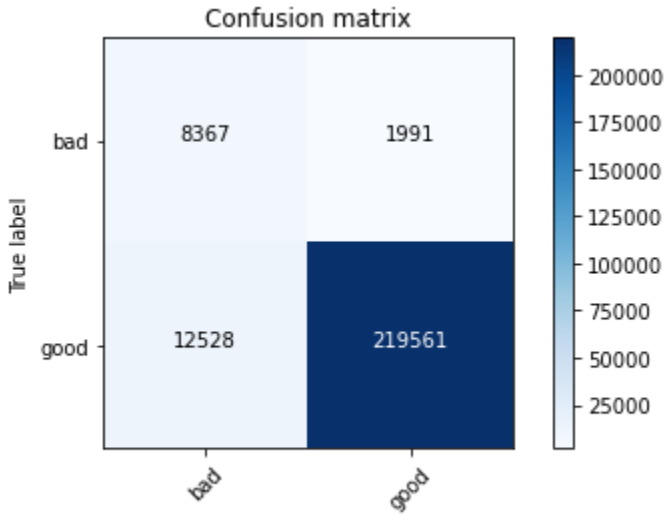
- better accuracy in labeling the bad probesets, misclassifying ~½ as many bad probesets as benchmark
- worse accuracy in labeling the good probesets, misclassifying ~2x as many good probesets as benchmark
- better balanced_accuracy



benchmark: Heuristic Classification
(derived from 'OriginalCT')

True	precision	recall	F-beta	support (n)
bad	0.531558	0.608998	0.567649	10358
good	0.982436	0.976048	0.979231	232089

balanced_accuracy: 79.3 %



Random Forest Classification

True	precision	recall	F-beta	support (n)
bad	0.400431	0.807781	0.535437	10358
good	0.991013	0.946021	0.967995	232089

balanced_accuracy: 87.7 %

SUMMARY

- balanced_accuracy of probeset classification improves with Random Forest model, relative to available strategy (“OriginalCT”)

FUTURE WORK

- confirm from subject matter experts whether balanced_accuracy is the appropriate performance metric to optimize
- understand likely sources of misclassifications: will additional feature engineering help?
- evaluate additional algorithms, like XG Boost
- evaluate additional data sets