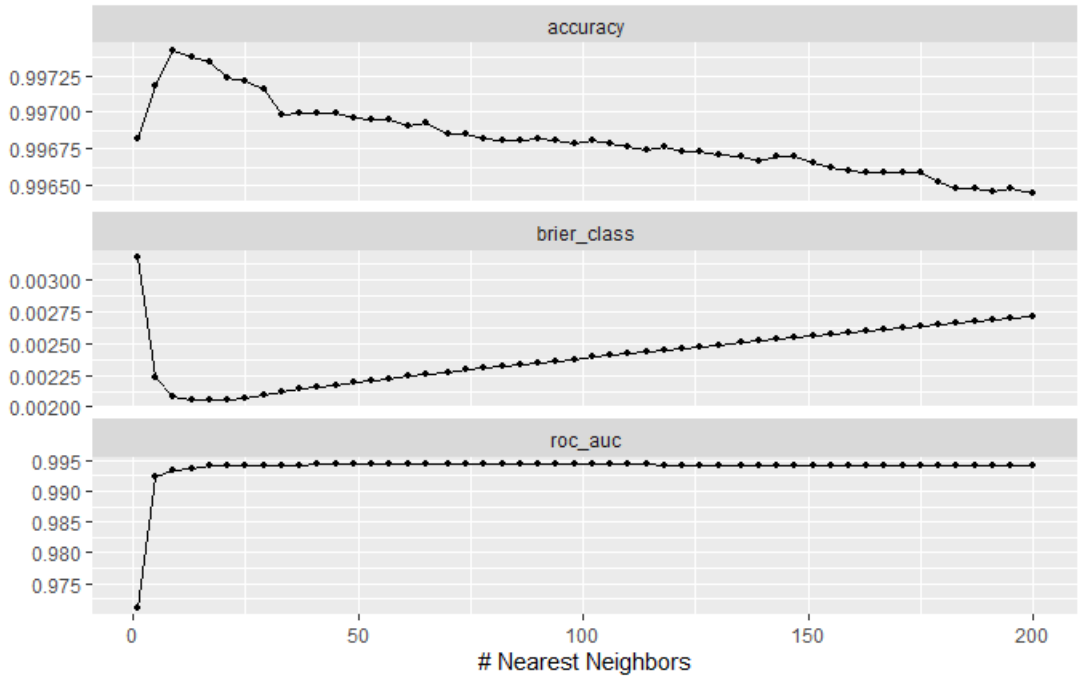
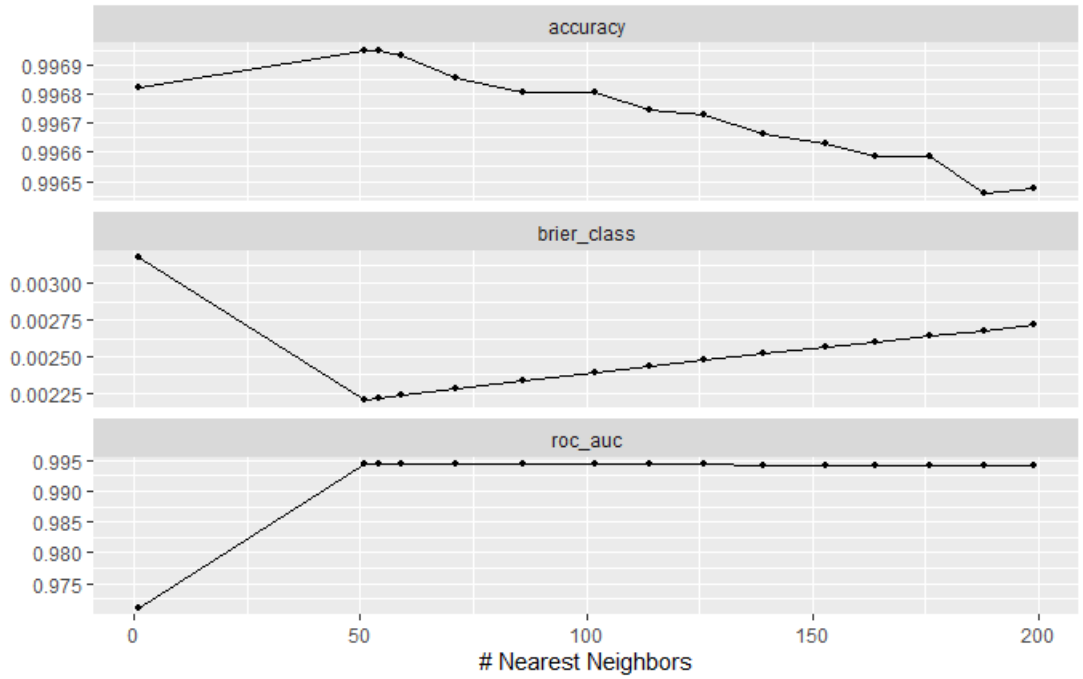
K-Nearest Neighbor Model _ full training set

Initial tuning neighbors (1~200)

tune_grid



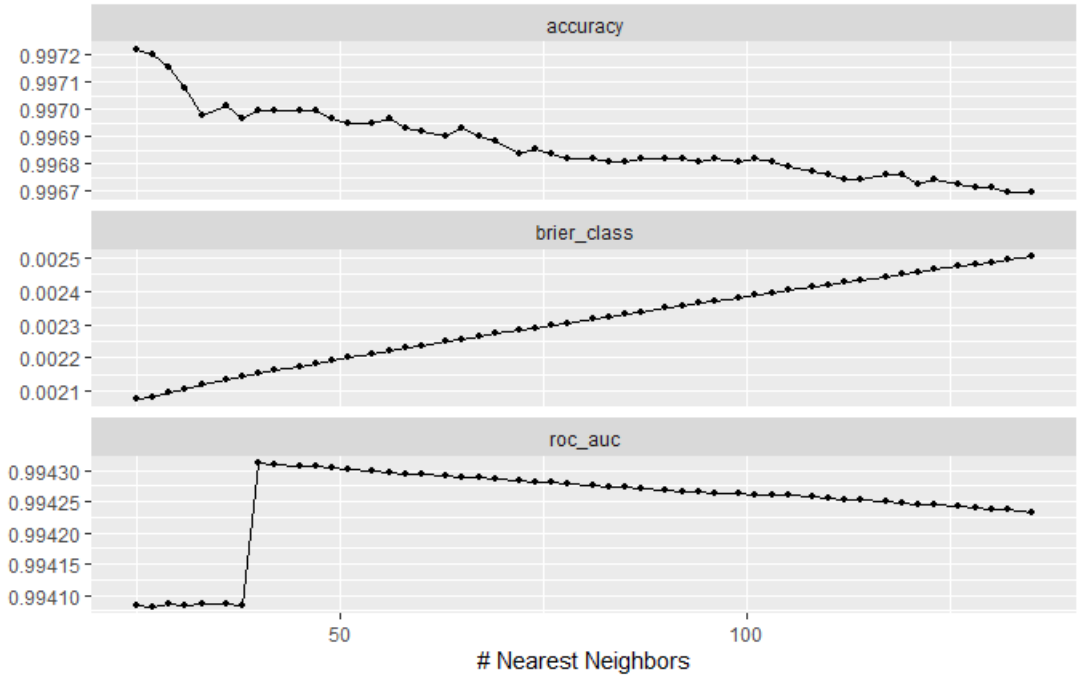
tune_bayes



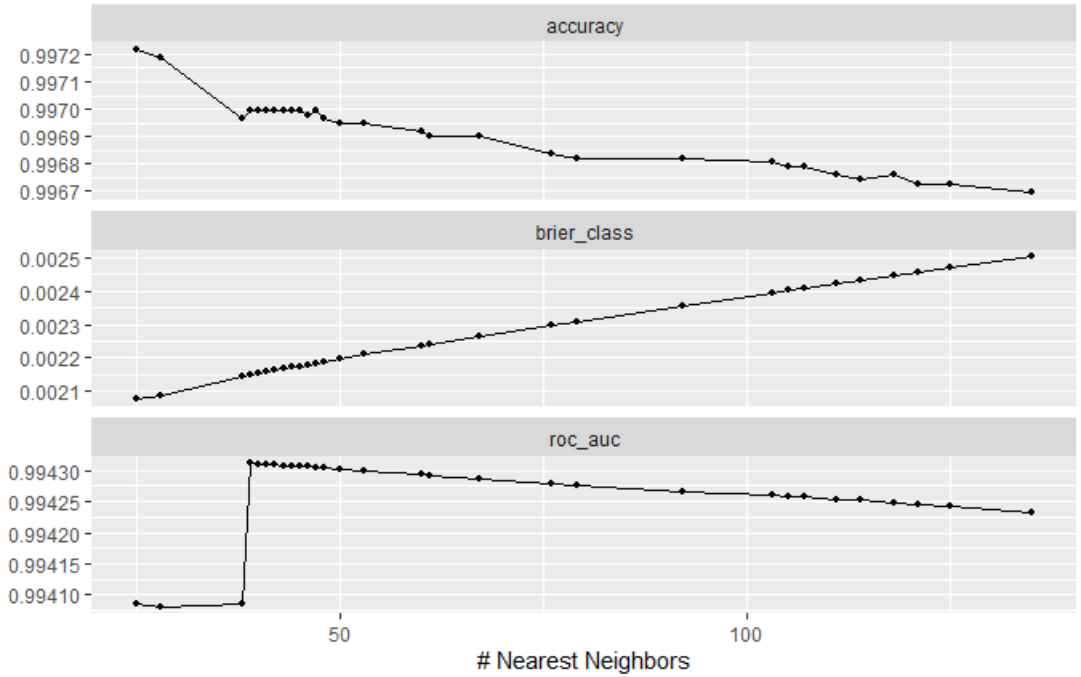
K-Nearest Neighbor Model _ full training set

Second tuning neighbors (25~135)

tune_grid

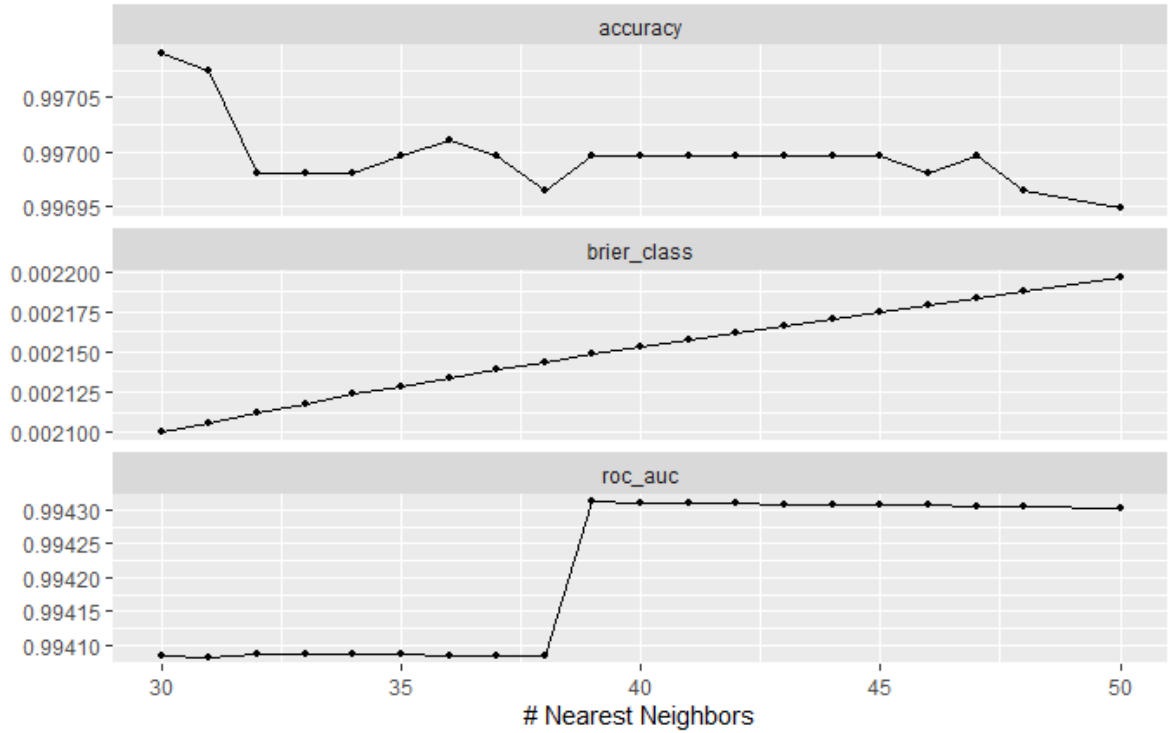


tune_bayes



K-Nearest Neighbor Model _ full training set

Final tuning neighbors (30~50)



A tibble: 3 × 7

neighbors <int>	.metric <chr>	.estimator <chr>	mean <dbl>	n <int>	std_err <dbl>	.config <chr>
39	roc_auc	binary	0.9943117	10	0.001139645	Preprocessor1_Model10
40	roc_auc	binary	0.9943110	10	0.001139833	Preprocessor1_Model11
41	roc_auc	binary	0.9943105	10	0.001139690	Preprocessor1_Model12

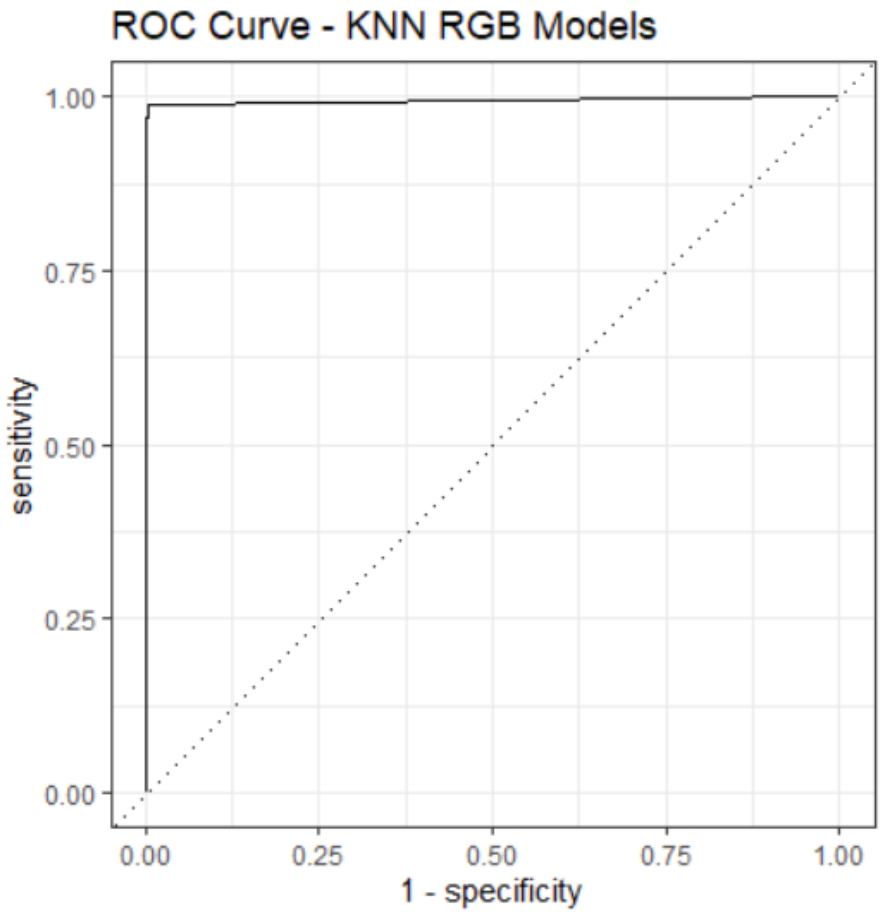
3 rows

K-Nearest Neighbor Model _ full training set

Fit the model with the optimal tuning parameter (neighbors=39) and estimate model performance with 10-fold cross validation

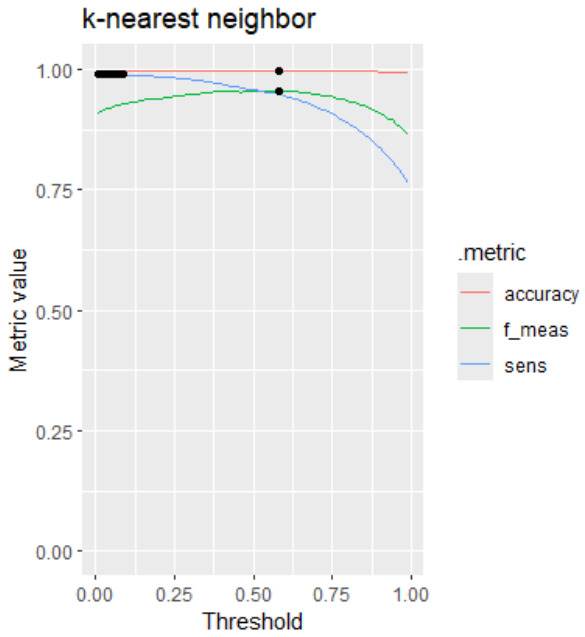
Cross-validation metrics for k-nearest neighbor models

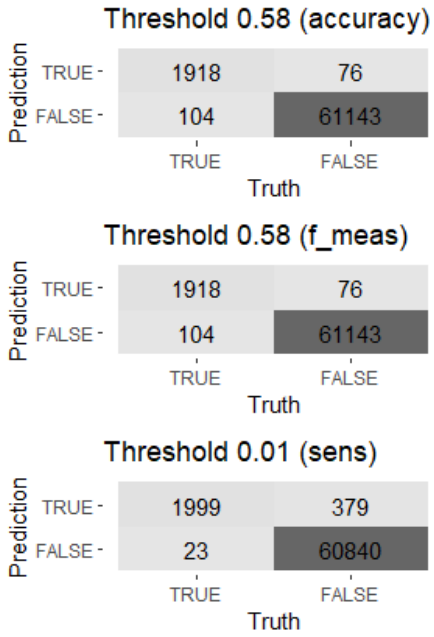
model	accuracy	f_meas	precision	roc_auc
k-nearest neighbor	0.997	0.953	0.949	0.994

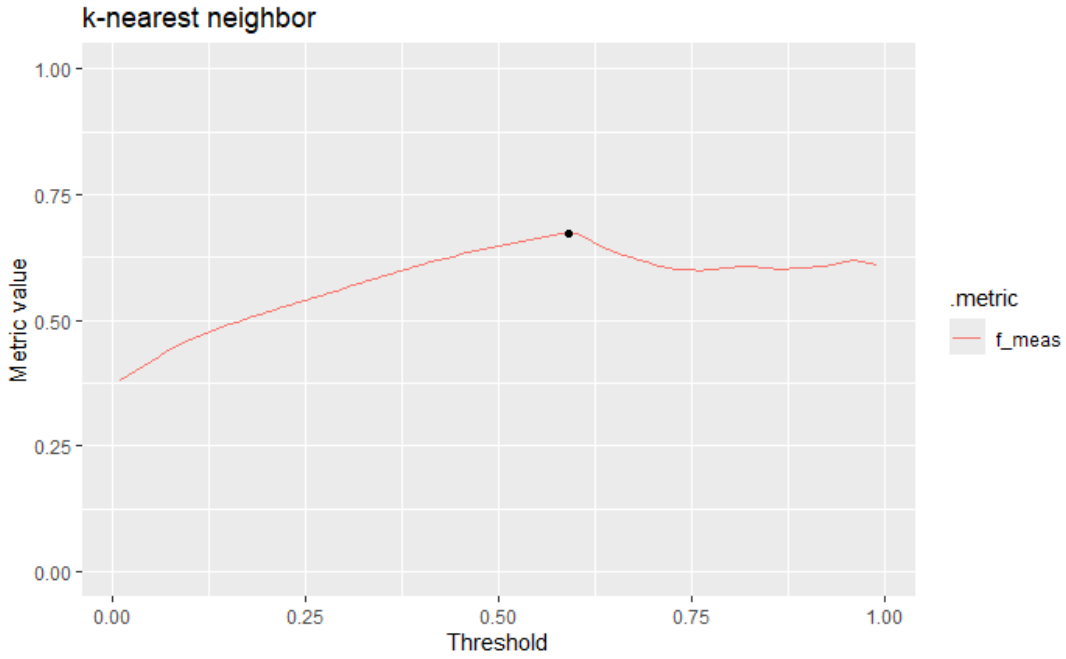


K-Nearest Neighbor Model _ full training set

Threshold scanning for both training (cross validation) and holdout data

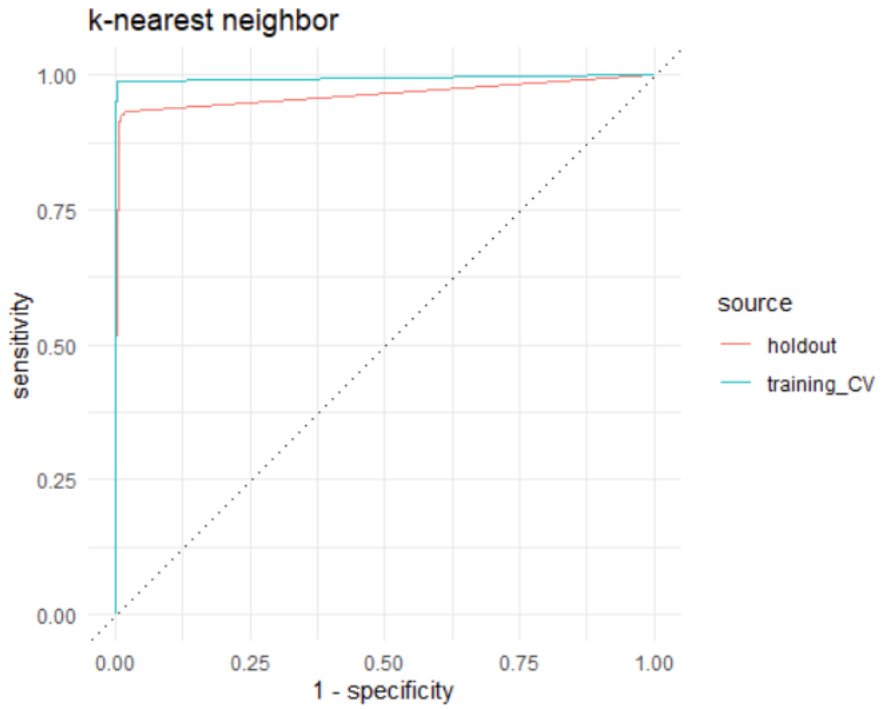






K-Nearest Neighbor Model _ full training set

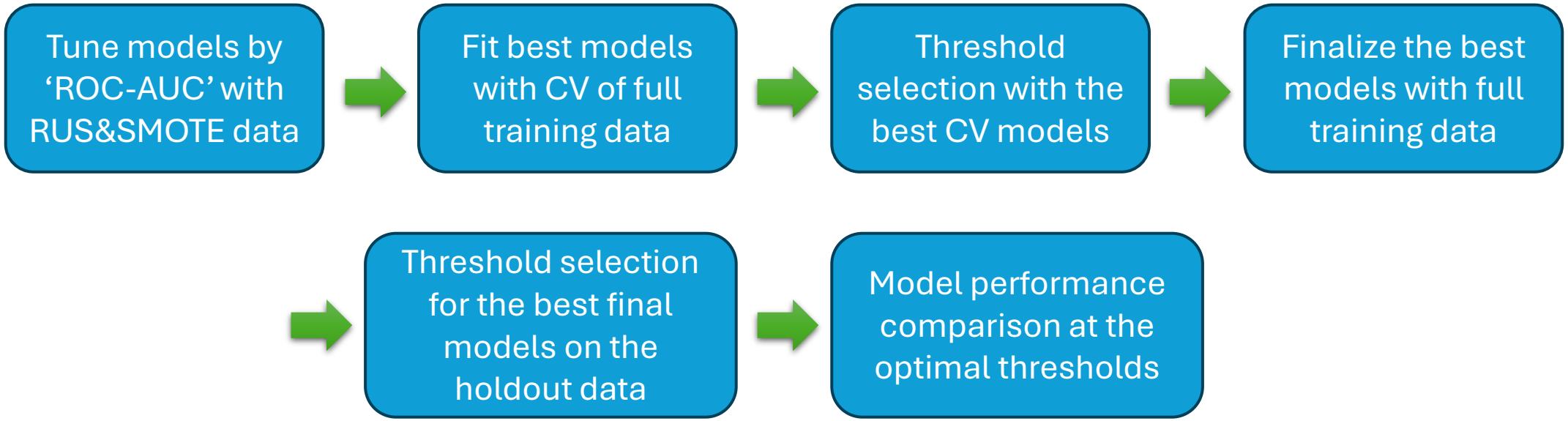
Comparison and performance of the tuned KNN model after threshold optimization



Performance metrics for tuned k-nearest neighbor model with optimal threshold

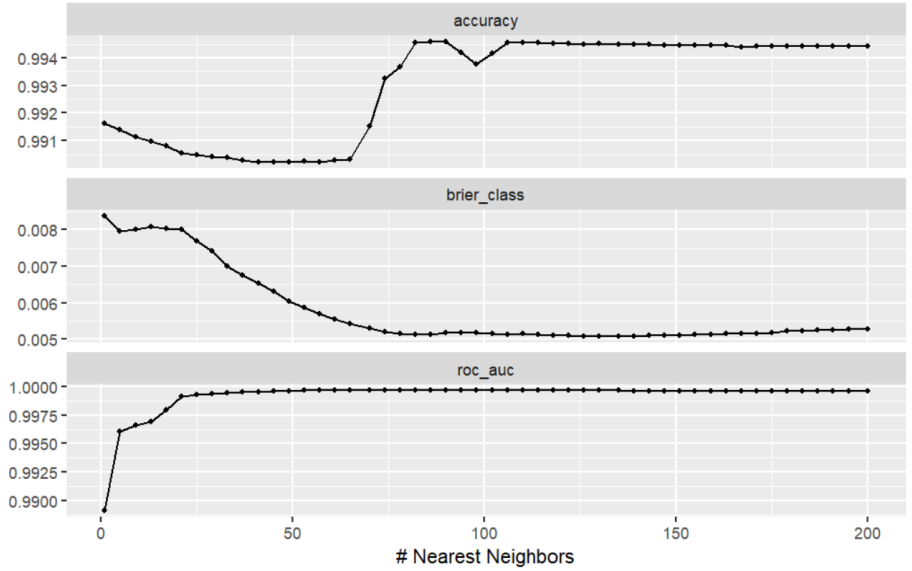
color_space	model	dataset	threshold	accuracy	kap	roc_auc	f_meas	sens
RGB	k-nearest neighbor	train	0.58	0.997	0.958	0.994	0.959	0.952
RGB	k-nearest neighbor	holdout	0.59	0.994	0.670	0.964	0.673	0.884

To lower the demand on computational resources for model tuning, we applied Random Under-sampling (RUS) combined with Synthetic Minority Over-sampling Technique (SMOTE) to pre-process the training data for model tuning and the following workflow for building KNN classifiers.

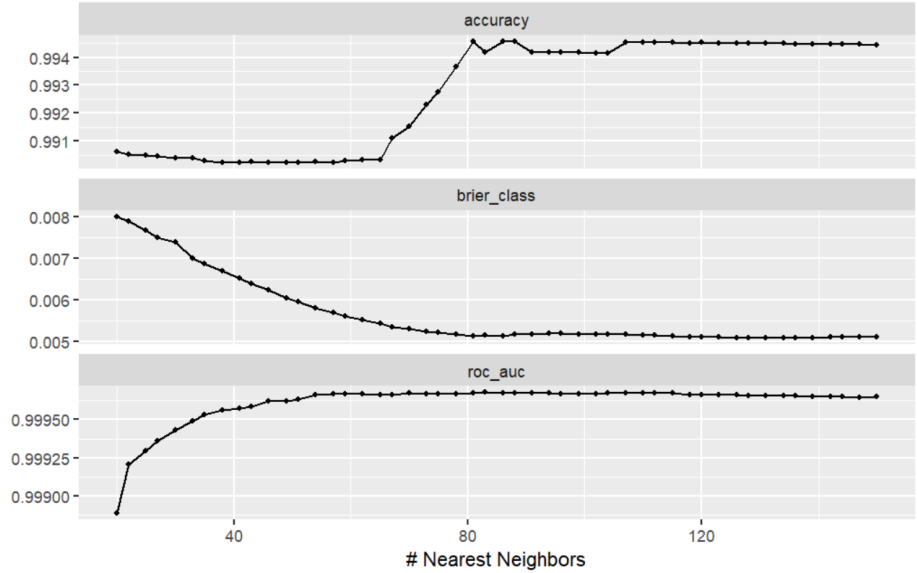


K-Nearest Neighbor Model _ RUS&SMOTE

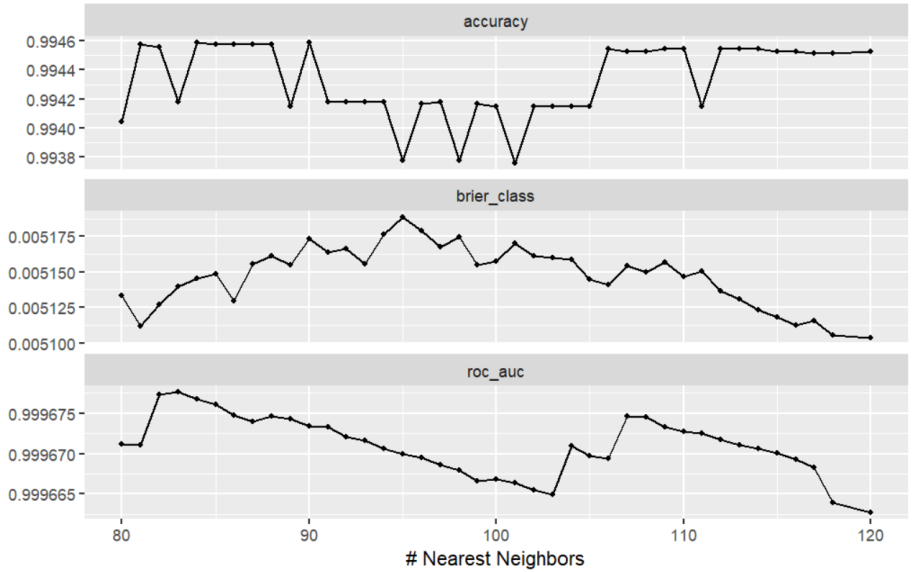
Initial tuning neighbors (1~200) *tune_grid*



Second tuning neighbors (20~150) *tune_grid*



Final tuning neighbors (80~120) *tune_grid*



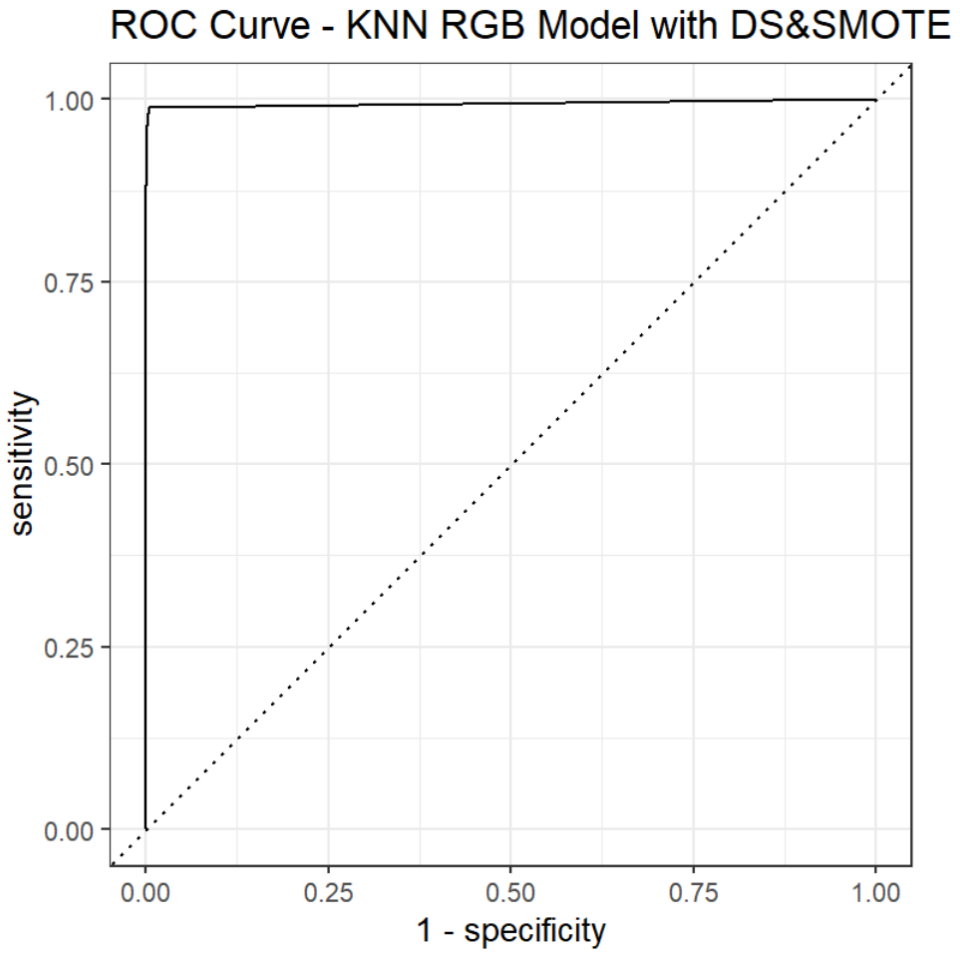
neighbors <int>	.metric <chr>	.estimator <chr>	mean <dbl>	n <int>	std_err <dbl>	.config <chr>
83	roc_auc	binary	0.9996776	10	3.057284e-05	Preprocessor1_Model04
82	roc_auc	binary	0.9996774	10	3.074929e-05	Preprocessor1_Model03
84	roc_auc	binary	0.9996767	10	3.077229e-05	Preprocessor1_Model05

K-Nearest Neighbor Model _ RUS&SMOTE

Fit the model with original workflow without down-sampling or SMOTE using the optimal tuning parameter (neighbors=83) and estimate model performance with 10-fold cross validation

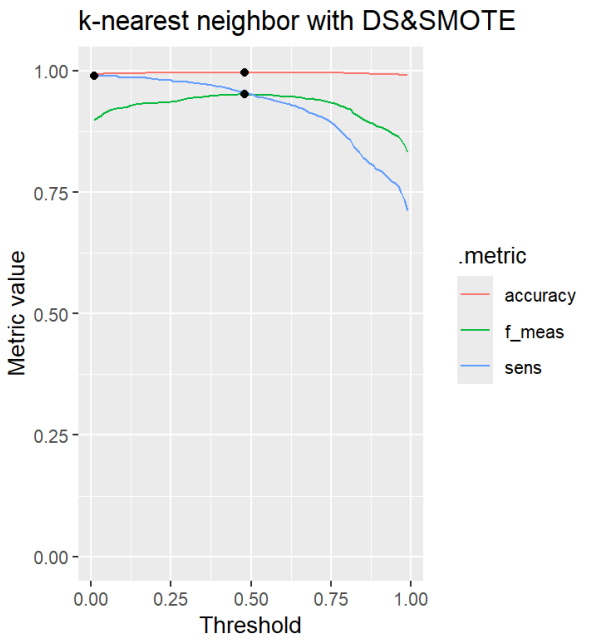
Cross-validation metrics for KNN models by Down-sample&SMOTE

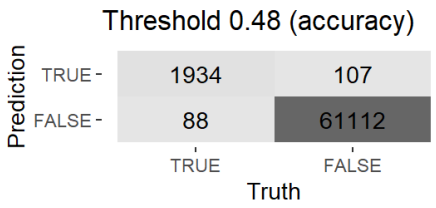
model	accuracy	f_meas	precision	roc_auc
Tuned KNN DS&SMOTE Model	0.997	0.95	0.95	0.994

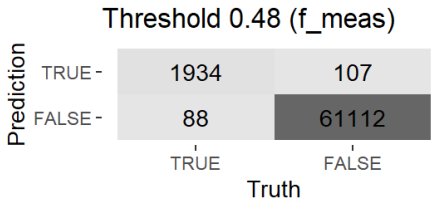


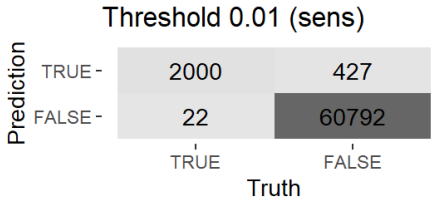
K-Nearest Neighbor Model _ RUS&SMOTE

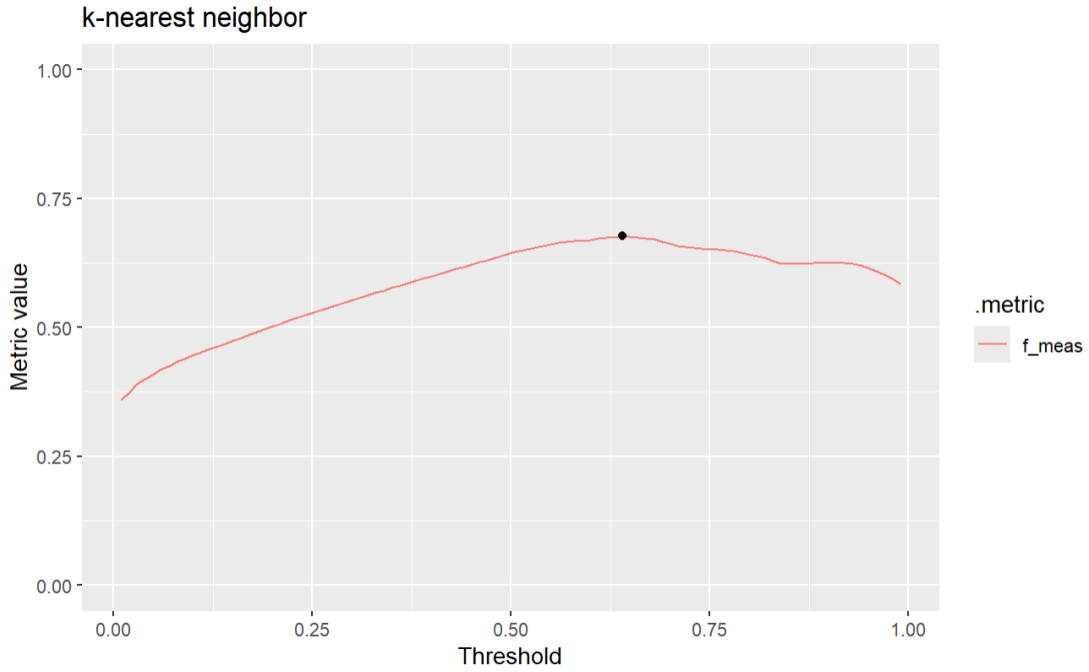
Threshold scanning for both training (cross validation) and holdout data





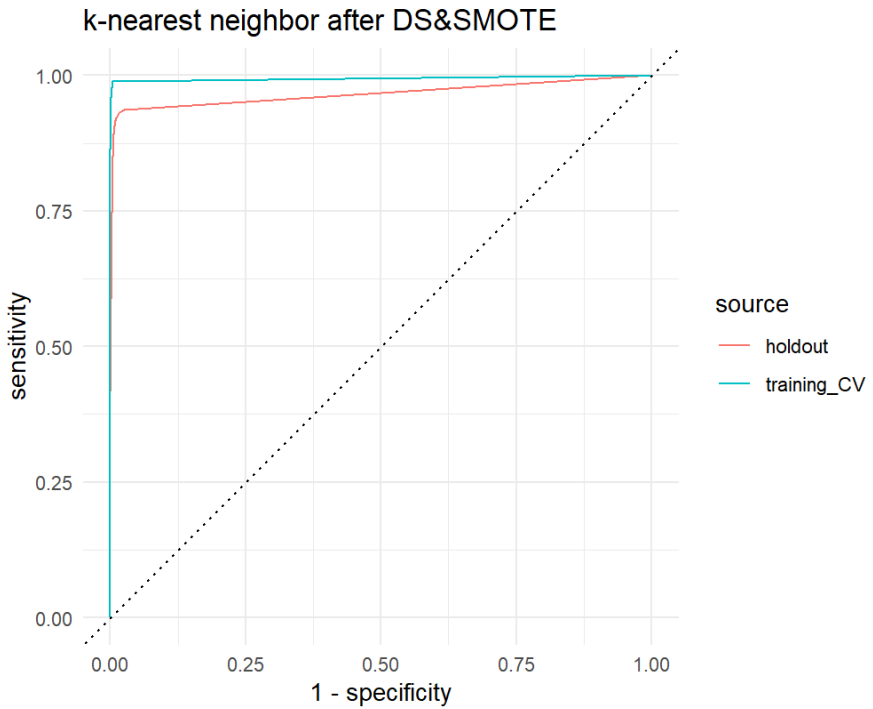






K-Nearest Neighbor Model _ RUS&SMOTE

Comparison and performance of the tuned KNN model after threshold optimization



Performance metrics for tuned KNN model after DS&SMOTE with optimal threshold

color_space	model	dataset	threshold	accuracy	kap	roc_auc	f_meas	sens
RGB	k-nearest neighbor after DS&SMOTE	train	0.48	0.997	0.953	0.994	0.954	0.958
RGB	k-nearest neighbor after DS&SMOTE	holdout	0.64	0.995	0.674	0.966	0.677	0.796

Evaluation of Random Under-sampling (RUS) combined with Synthetic Minority Over-sampling Technique (SMOTE) on the KNN model performances.

Performance metrics for tuned k-nearest neighbor model with optimal threshold

color_space	model	dataset	threshold	accuracy	kap	roc_auc	f_meas	sens
RGB	k-nearest neighbor	train	0.58	0.997	0.958	0.994	0.959	0.952
RGB	k-nearest neighbor	holdout	0.59	0.994	0.670	0.964	0.673	0.884

Performance metrics for tuned KNN model after DS&SMOTE with optimal threshold

color_space	model	dataset	threshold	accuracy	kap	roc_auc	f_meas	sens
RGB	k-nearest neighbor after DS&SMOTE	train	0.48	0.997	0.953	0.994	0.954	0.958
RGB	k-nearest neighbor after DS&SMOTE	holdout	0.64	0.995	0.674	0.966	0.677	0.796

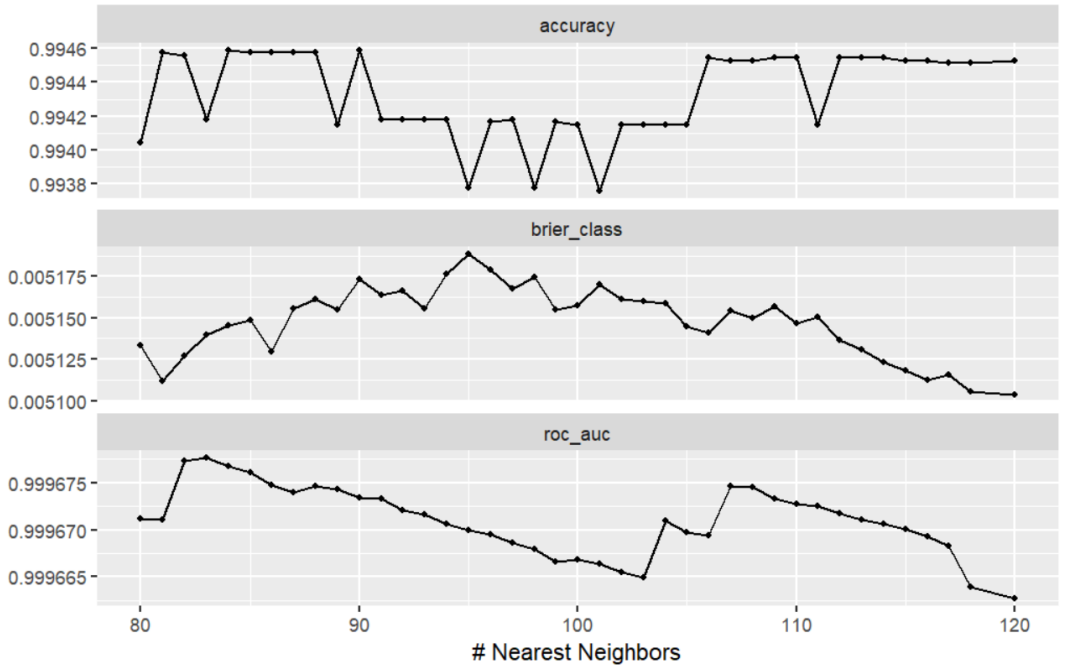
The strategy of Random Under-sampling (RUS) combined with Synthetic Minority Over-sampling Technique (SMOTE) even improved model performance a little on the holdout data.

Based on the result, the same hybrid pre-processing method (RUS&SMOTE) was applied to the tuning (only tuning) of all the models built here and after.

K-Nearest Neighbor Model - RGB vs. Engineered Features

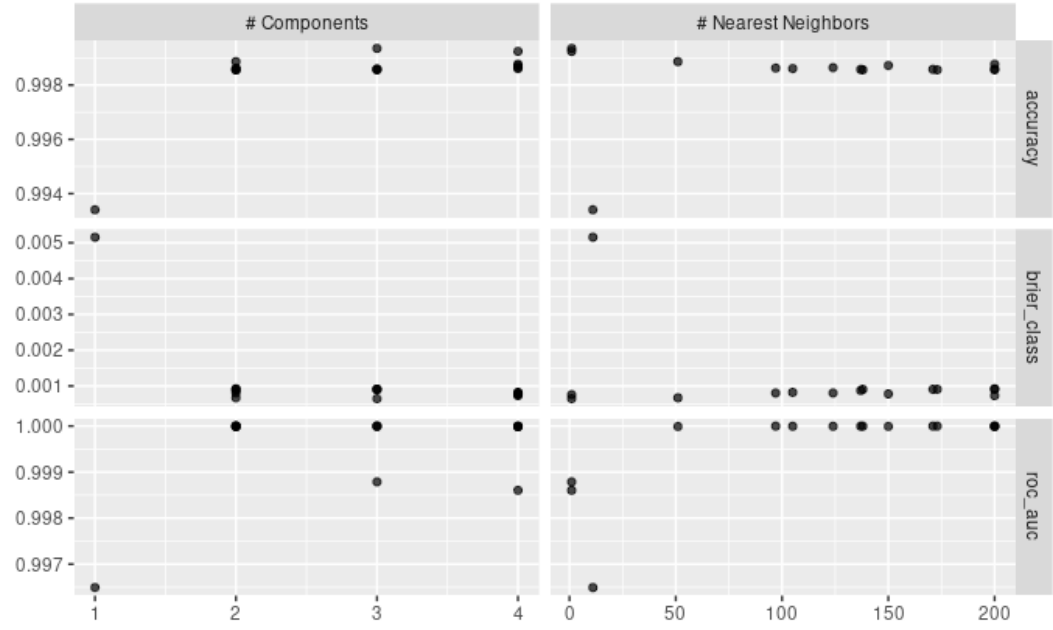
Comparison between the RGB feature space and engineered full feature space for KNN models

RGB



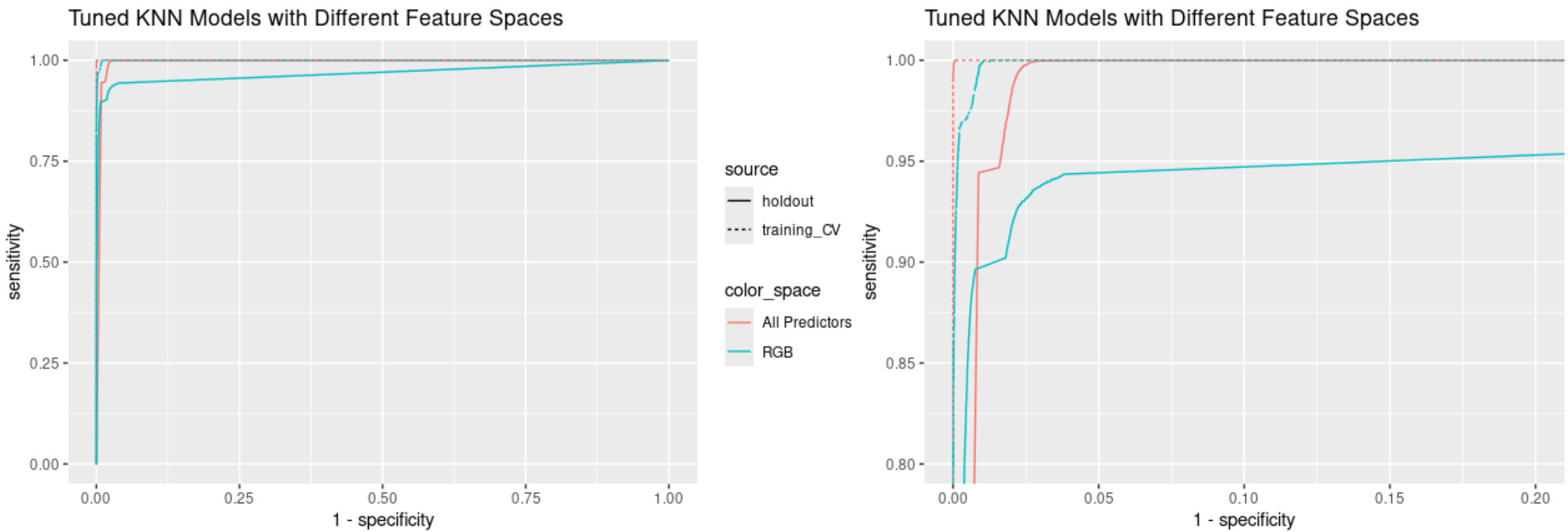
neighbors <int>	.metric <chr>	.estimator <chr>	mean <dbl>	n <int>	std_err <dbl>
83	roc_auc	binary	0.9996776	10	3.057284e-05
82	roc_auc	binary	0.9996774	10	3.074929e-05
84	roc_auc	binary	0.9996767	10	3.077229e-05

All Engineered Features + PCA



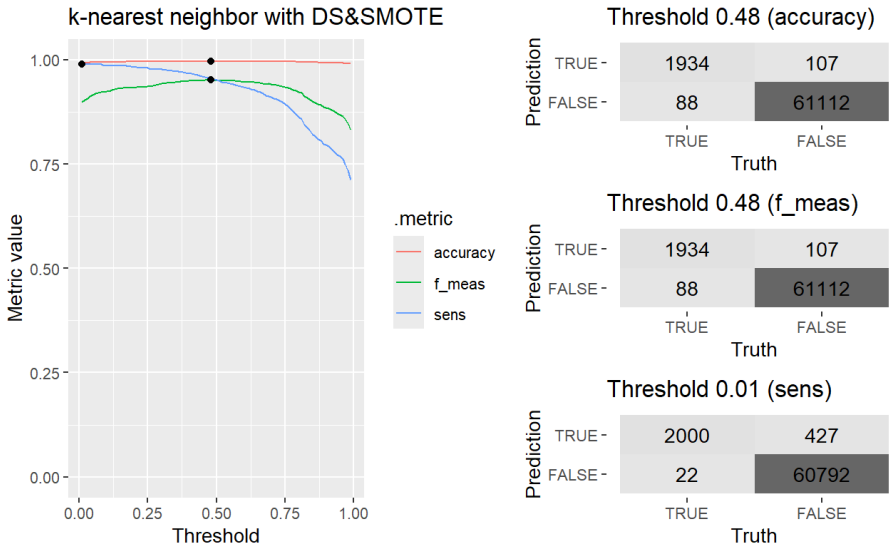
neighbors <int>	num_comp <int>	.metric <chr>	.estimator <chr>	mean <dbl>	n <int>	std_err <dbl>
173	2	roc_auc	binary	0.9999974	10	1.349389e-06
200	3	roc_auc	binary	0.9999974	10	1.196654e-06
137	2	roc_auc	binary	0.9999974	10	1.310655e-06

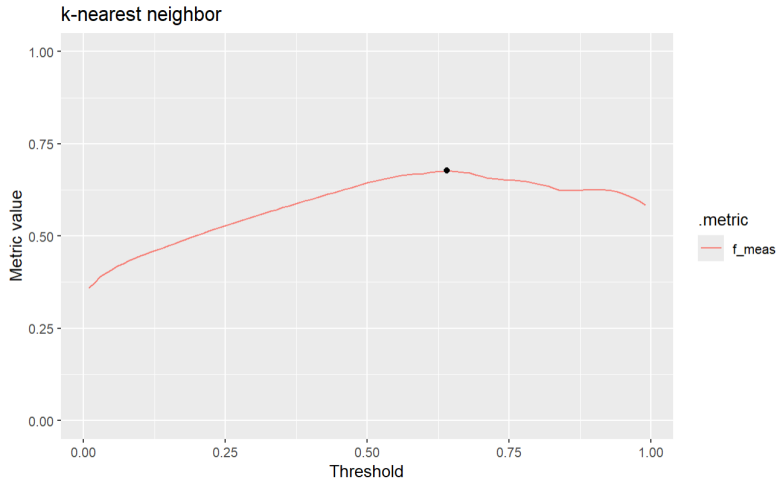
Comparison between the RGB feature space and engineered full feature space for KNN models



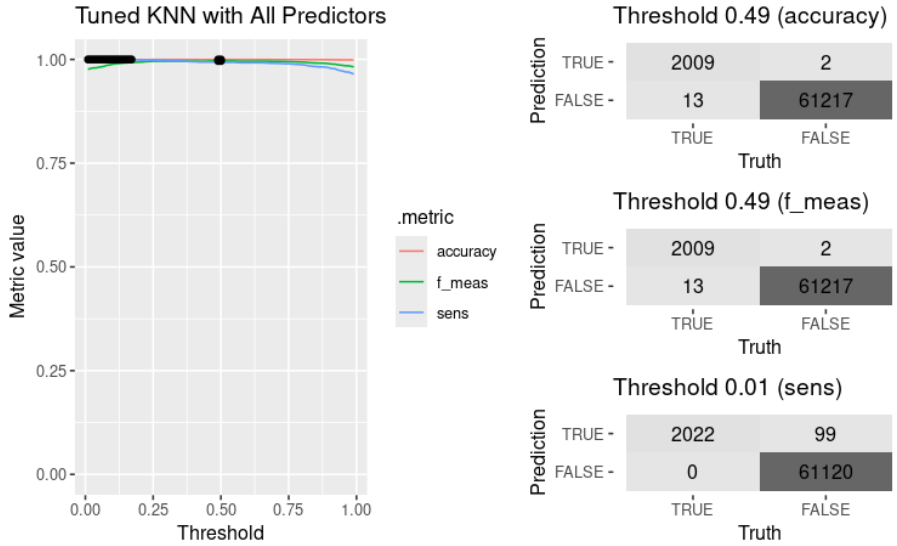
Comparison between the RGB feature space and engineered full feature space for KNN models

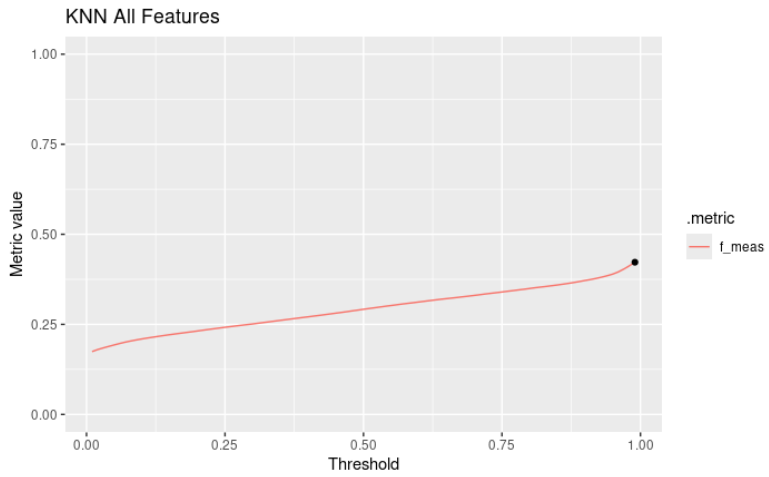
RGB





All Engineered Features + PCA





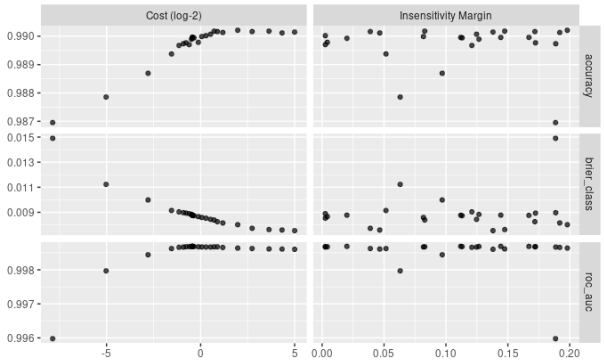
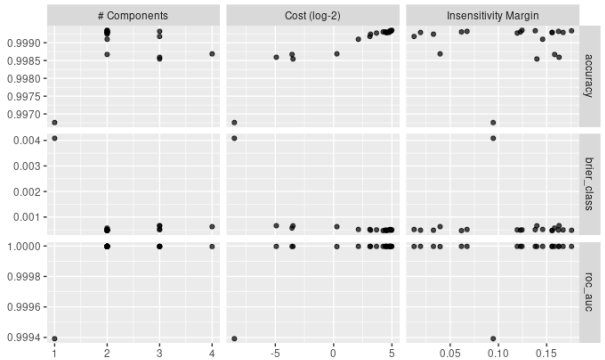
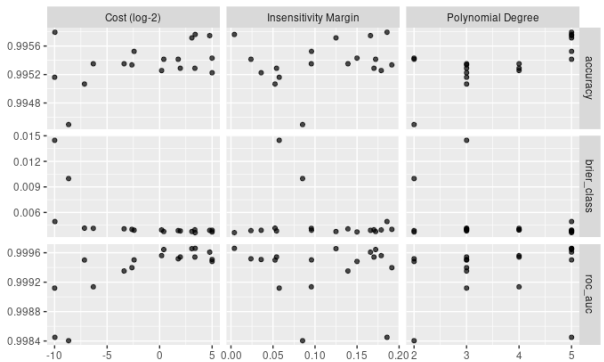
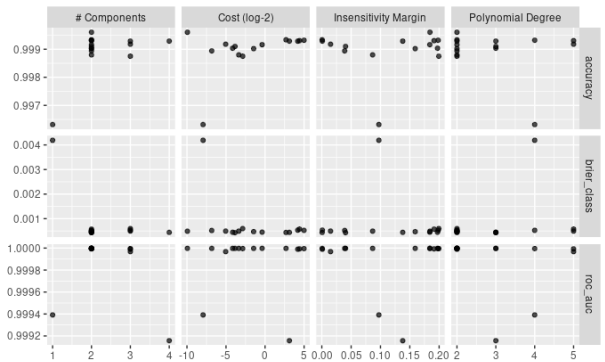
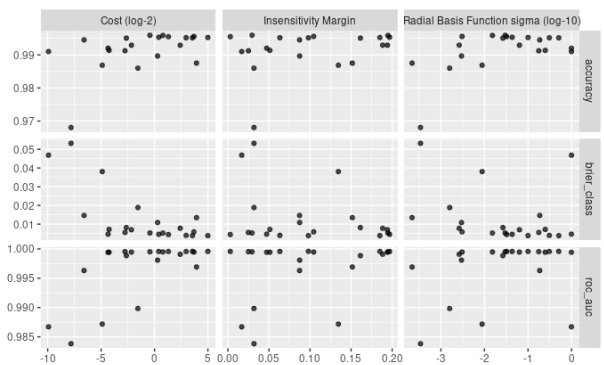
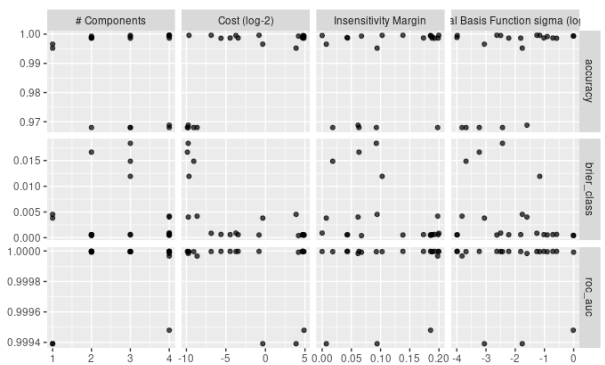
Comparison between the RGB feature space and engineered full feature space for KNN models

Performance metrics for tuned KNN model at optimal thresholds

color_space	model	dataset	threshold	accuracy	kap	f_meas	roc_auc
RGB	k-nearest neighbor	train	0.49	0.997	0.952	0.954	1.000
RGB	k-nearest neighbor	holdout	0.64	0.995	0.682	0.685	0.968

Build SVM classifiers

Build and Tune the SVM Models in Different Feature Spaces

	RGB	All Engineered Features + PCA
Linear kernel		
Polynomial kernel		
Radial basis function kernel		

Cross-validation Performance of the SVM Models Build in Different Feature Spaces

RGB

Cross-validation performance metrics

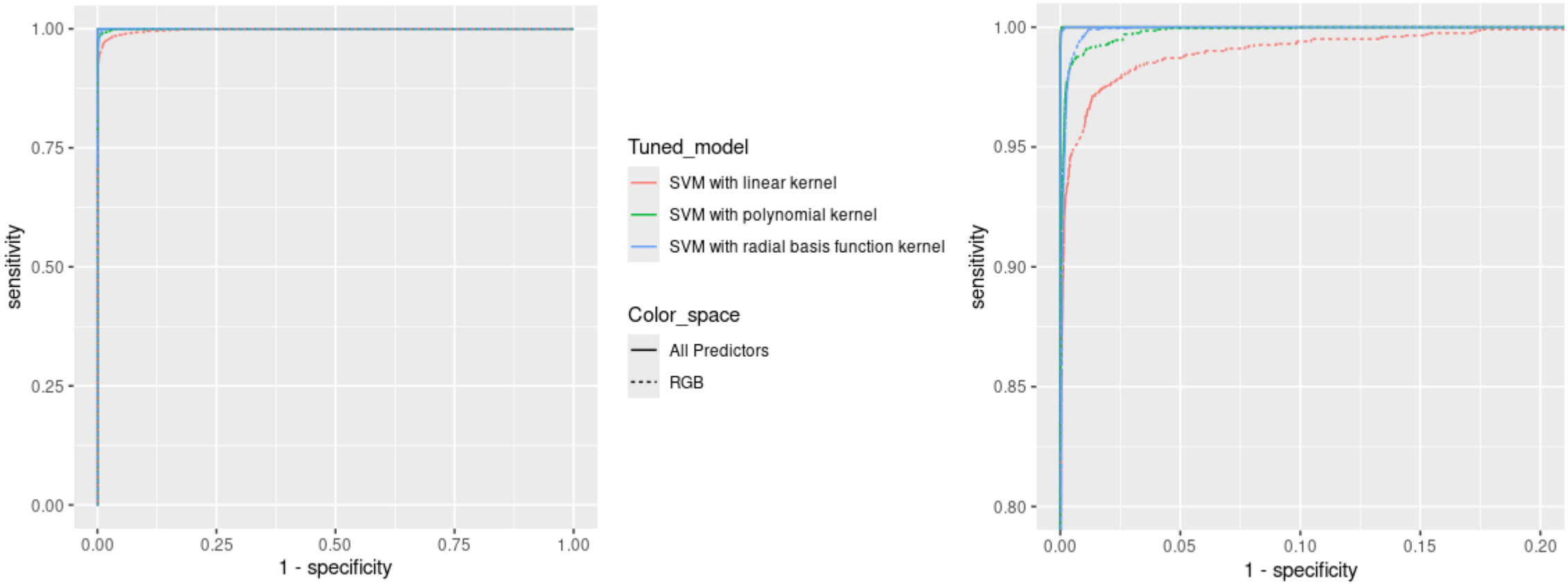
model	accuracy	f_meas	precision	roc_auc
SVM with linear kernel	0.995	0.915	0.976	0.998
SVM with polynomial kernel	0.997	0.955	0.959	1.000
SVM with radial basis function kernel	0.997	0.947	0.950	1.000

All Engineered Features + PCA

Cross-validation performance metrics for SVM models with all feature space

model	accuracy	f_meas	precision	roc_auc
SVM with linear kernel	1.000	0.995	0.997	1
SVM with polynomial kernel	1.000	0.996	0.998	1
SVM with radial basis function kernel	0.999	0.991	1.000	1

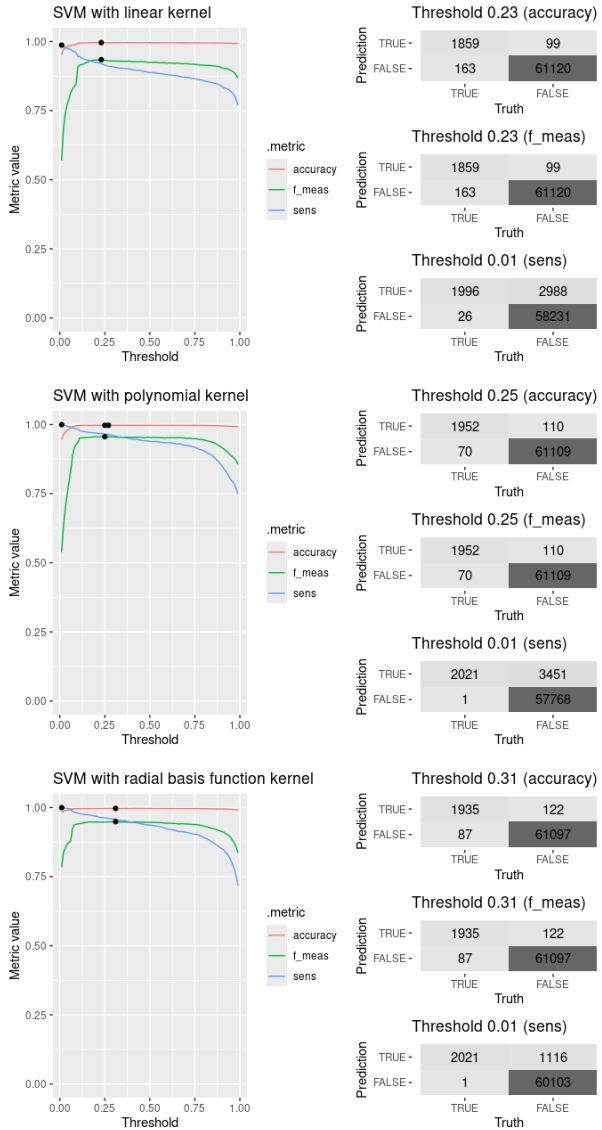
ROC curve of all SVM models with training CV



Threshold Selection for the SVM Models Build in Different Feature Spaces (with cross-validation)

RGB

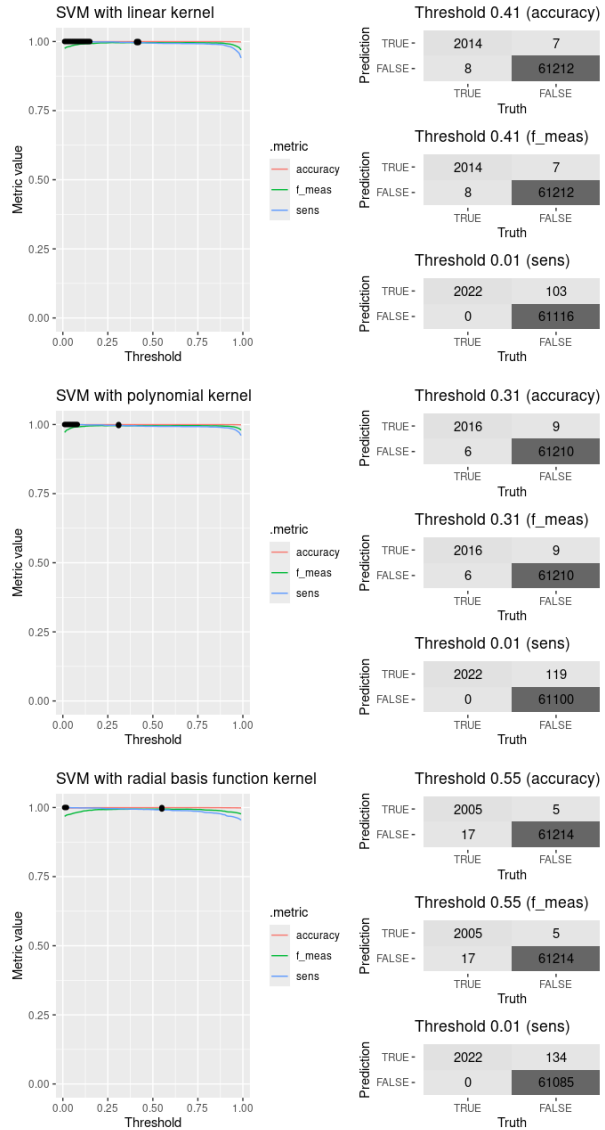
Linear kernel



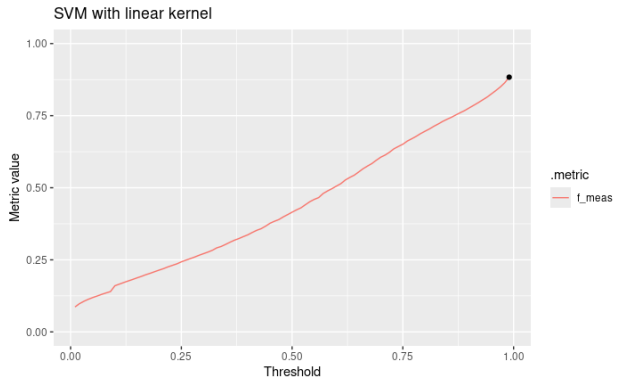
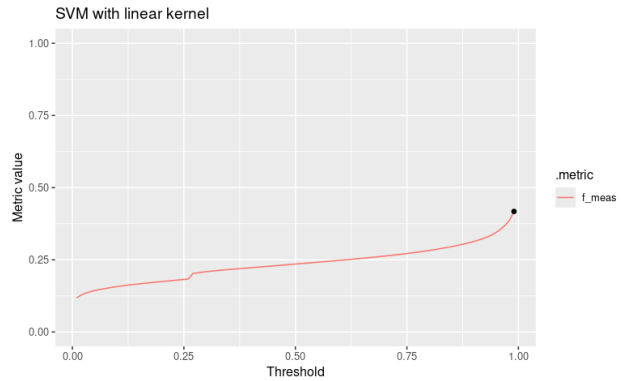
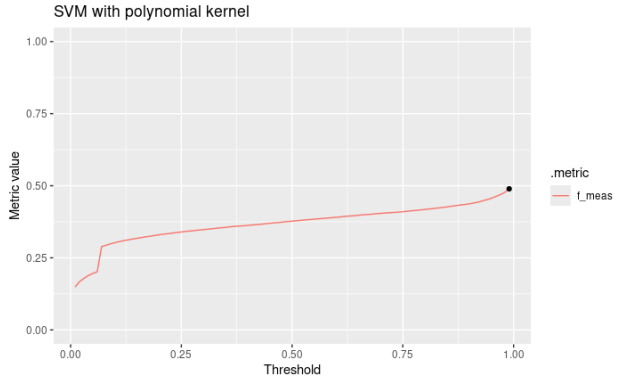
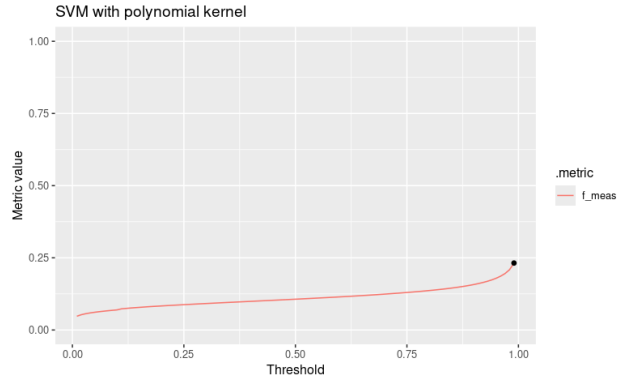
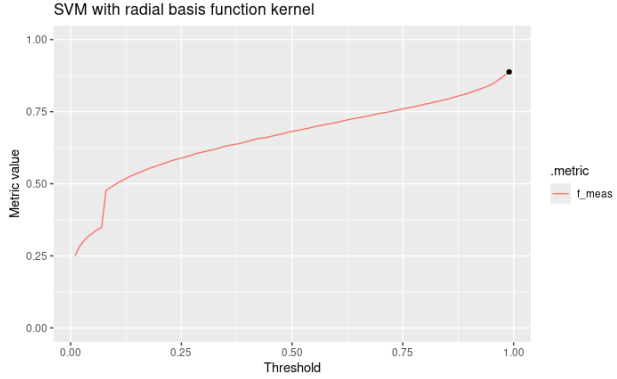
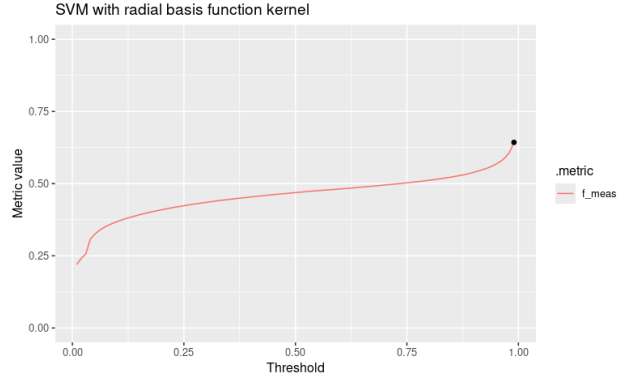
Polynomial kernel

Radial basis function kernel

All Engineered Features + PCA

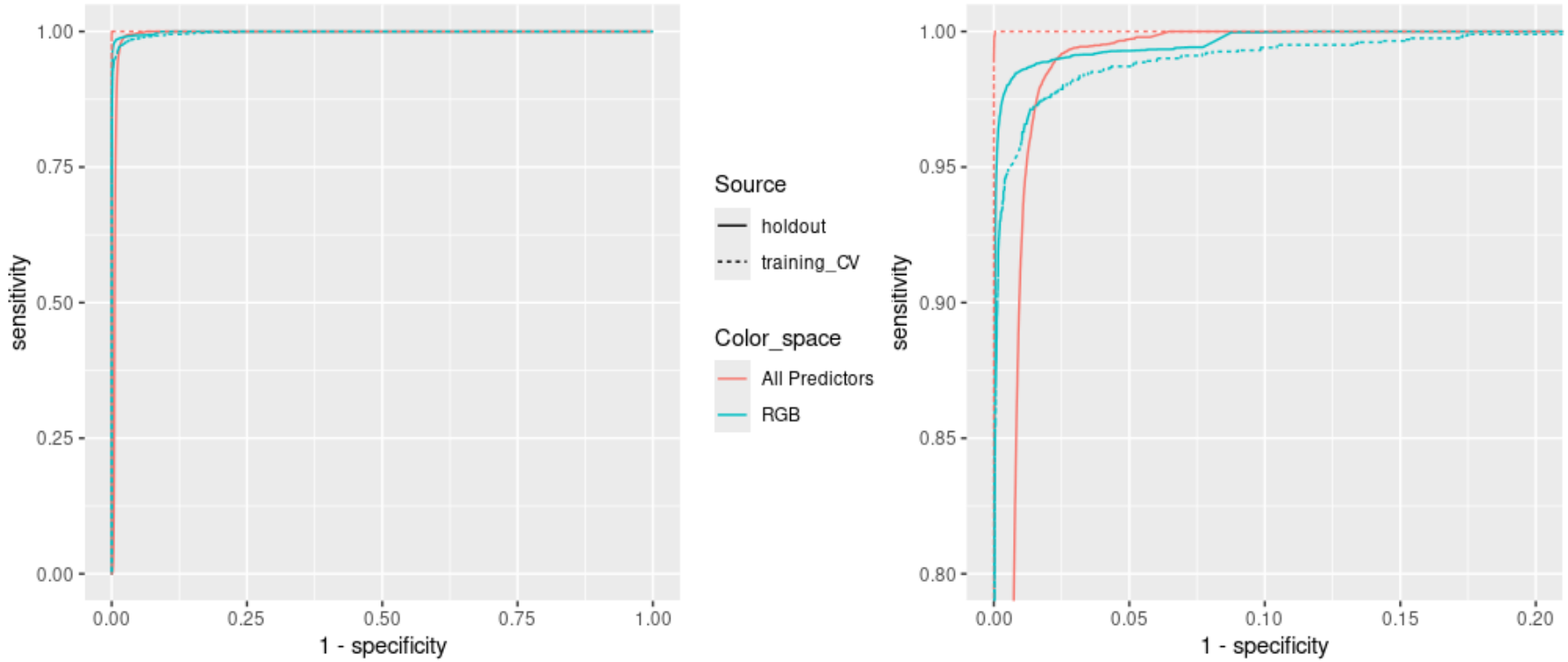


Threshold Selection for the SVM Models Build in Different Feature Spaces (with holdout data)

	RGB	**All Engineered Features + PCA**
Linear kernel		
Polynomial kernel		
Radial basis function kernel		

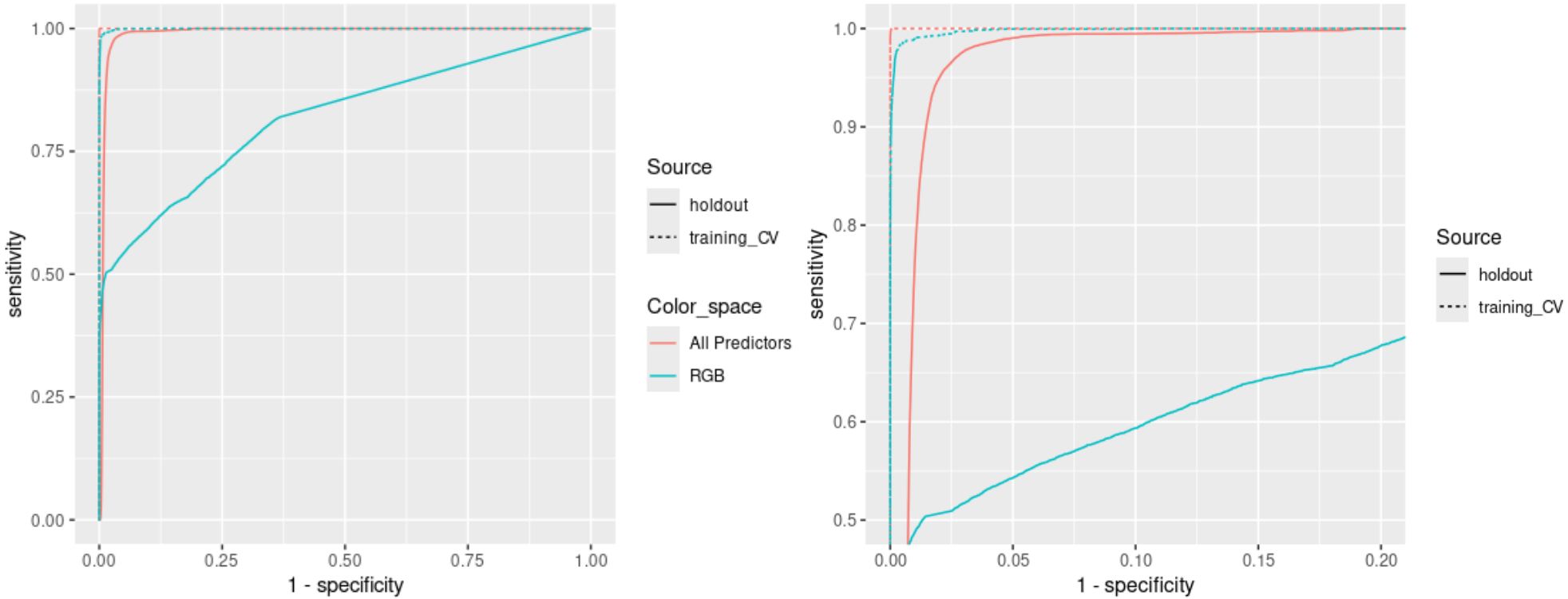
Comparison between SVM Models for Each Kernel in Different Feature Spaces

Linear kernel



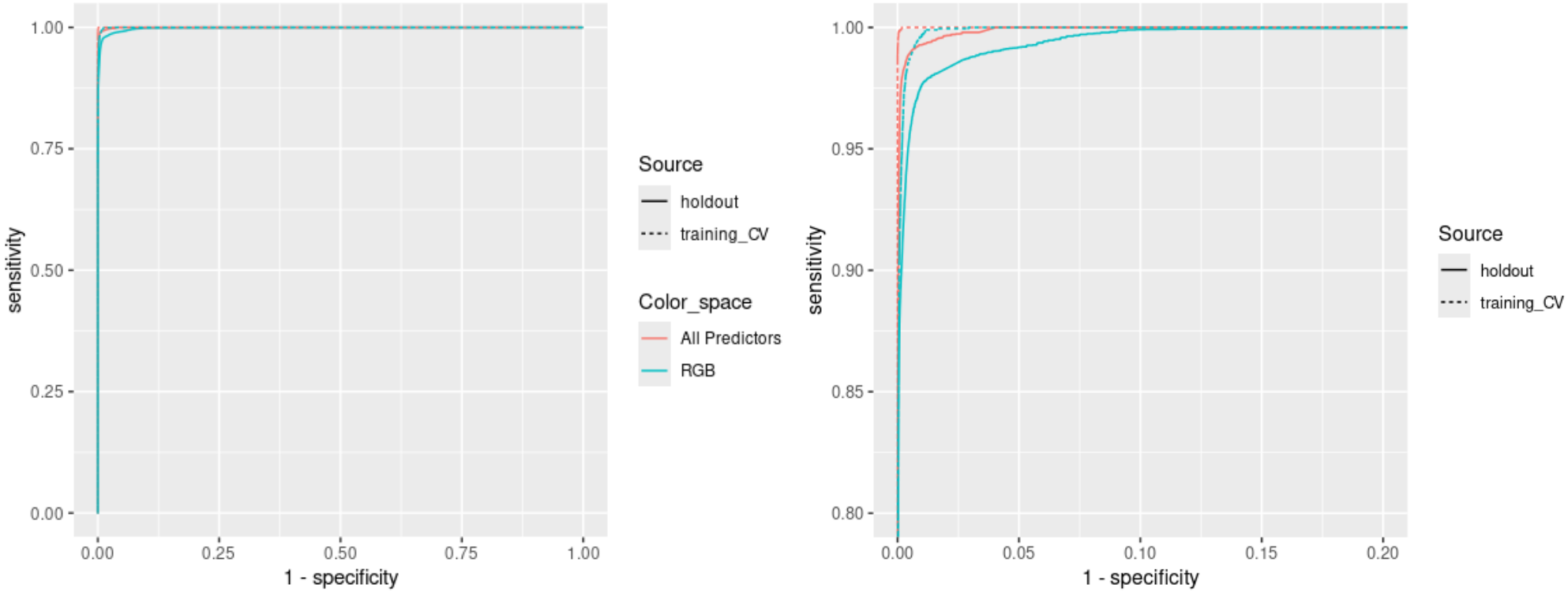
Comparison between SVM Models for Each Kernel in Different Feature Spaces

Polynomial kernel



Comparison between SVM Models for Each Kernel in Different Feature Spaces

Radial basis function kernel



Performance Metrics for All of the SVM Models Built in This Work

Performance Metrics for Tuned SVM Models at Optimal Threshold

color_space	model	dataset	threshold	accuracy	kap	f_meas	roc_auc
All Predictors	SVM with linear kernel	train	0.41	1.000	0.996	0.996	1.000
All Predictors	SVM with linear kernel	holdout	0.99	0.980	0.411	0.418	0.994
All Predictors	SVM with polynomial kernel	train	0.31	1.000	0.996	0.997	1.000
All Predictors	SVM with polynomial kernel	holdout	0.99	0.953	0.222	0.232	0.990
All Predictors	SVM with radial basis function kernel	train	0.55	1.000	0.995	0.995	1.000
All Predictors	SVM with radial basis function kernel	holdout	0.99	0.992	0.639	0.643	1.000
RGB	SVM with linear kernel	train	0.23	0.996	0.932	0.934	0.998
RGB	SVM with linear kernel	holdout	0.99	0.998	0.883	0.884	0.999
RGB	SVM with polynomial kernel	train	0.25	0.997	0.956	0.957	1.000
RGB	SVM with polynomial kernel	holdout	0.99	0.994	0.486	0.489	0.819
RGB	SVM with radial basis function kernel	train	0.31	0.997	0.947	0.949	1.000
RGB	SVM with radial basis function kernel	holdout	0.99	0.998	0.887	0.888	0.999