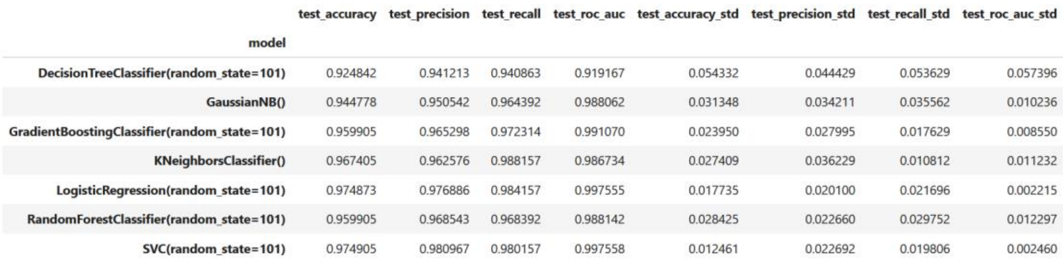
Comparison of different ML models and different sets of features

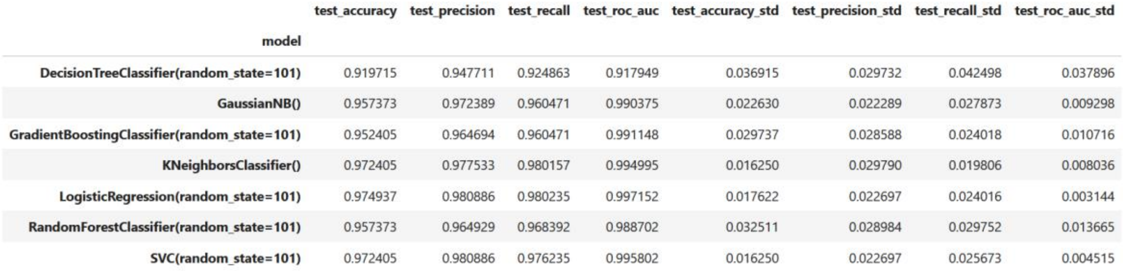
(Mean and SD values are calculated from the results of cross-validation)

1. All features

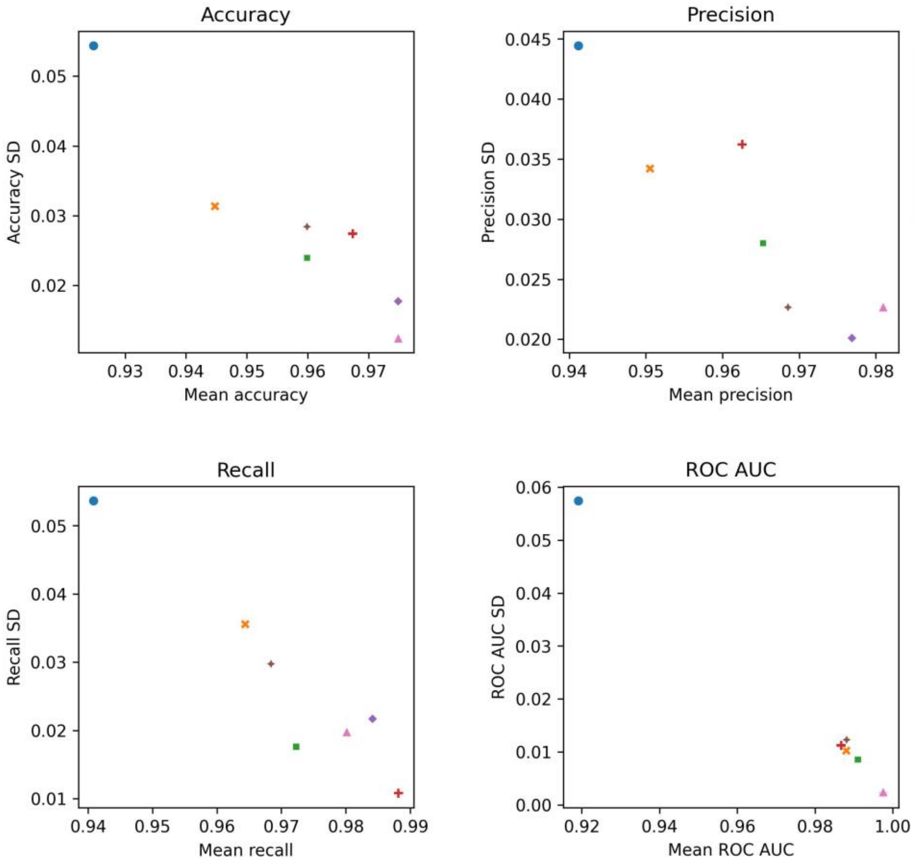


model	test_accuracy	test_precision	test_recall	test_roc_auc	test_accuracy_std	test_precision_std	test_recall_std	test_roc_auc_std
DecisionTreeClassifier(random_state=101)	0.924842	0.941213	0.940863	0.919167	0.054332	0.044429	0.053629	0.057396
GaussianNB()	0.944778	0.950542	0.964392	0.988062	0.031348	0.034211	0.035562	0.010236
GradientBoostingClassifier(random_state=101)	0.959905	0.965298	0.972314	0.991070	0.023950	0.027995	0.017629	0.008550
KNeighborsClassifier()	0.967405	0.962576	0.988157	0.986734	0.027409	0.036229	0.010812	0.011232
LogisticRegression(random_state=101)	0.974873	0.976886	0.984157	0.997555	0.017735	0.020100	0.021696	0.002215
RandomForestClassifier(random_state=101)	0.959905	0.968543	0.968392	0.988142	0.028425	0.022660	0.029752	0.012297
SVC(random_state=101)	0.974905	0.980967	0.980157	0.997558	0.012461	0.022692	0.019806	0.002460

2. Top 10 features

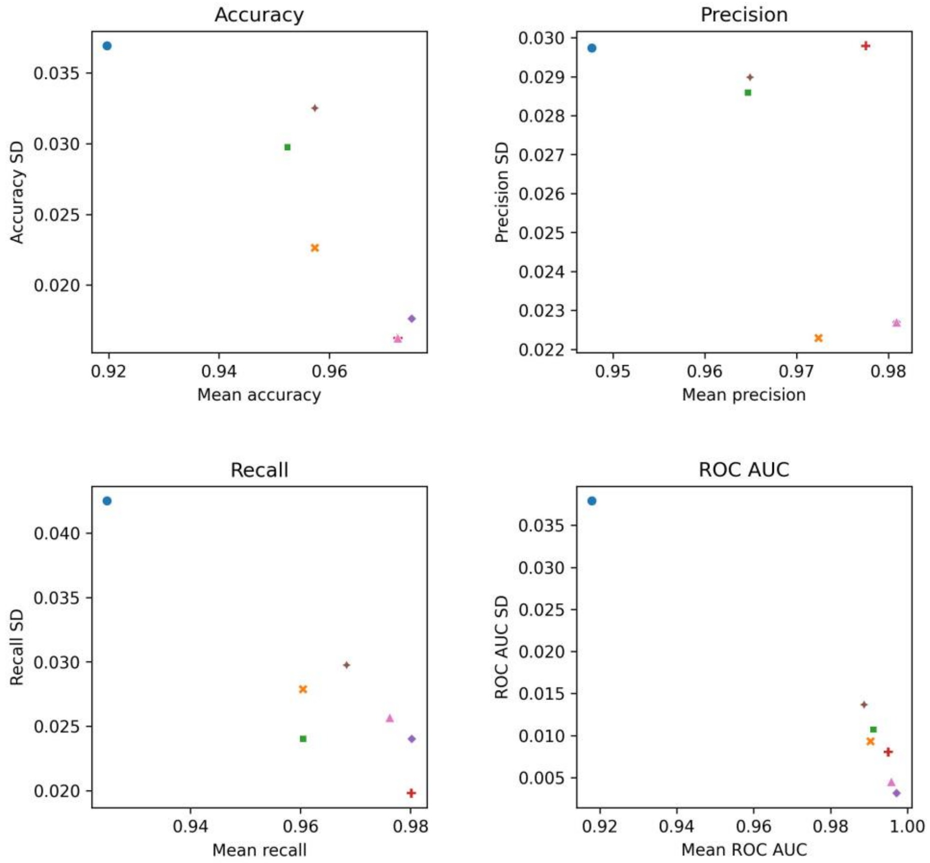


model	test_accuracy	test_precision	test_recall	test_roc_auc	test_accuracy_std	test_precision_std	test_recall_std	test_roc_auc_std
DecisionTreeClassifier(random_state=101)	0.919715	0.947711	0.924863	0.917949	0.036915	0.029732	0.042498	0.037896
GaussianNB()	0.957373	0.972389	0.960471	0.990375	0.022630	0.022289	0.027873	0.009298
GradientBoostingClassifier(random_state=101)	0.952405	0.964694	0.960471	0.991148	0.029737	0.028588	0.024018	0.010716
KNeighborsClassifier()	0.972405	0.977533	0.980157	0.994995	0.016250	0.029790	0.019806	0.008036
LogisticRegression(random_state=101)	0.974937	0.980886	0.980235	0.997152	0.017622	0.022697	0.024016	0.003144
RandomForestClassifier(random_state=101)	0.957373	0.964929	0.968392	0.988702	0.032511	0.028984	0.029752	0.013665
SVC(random_state=101)	0.972405	0.980886	0.976235	0.995802	0.016250	0.022697	0.025673	0.004515



● DecisionTreeClassifier(random_state=101)
× GaussianNB()
■ GradientBoostingClassifier(random_state=101)
+ KNeighborsClassifier()
◆ LogisticRegression(random_state=101)
+ RandomForestClassifier(random_state=101)
▲ SVC(random_state=101)

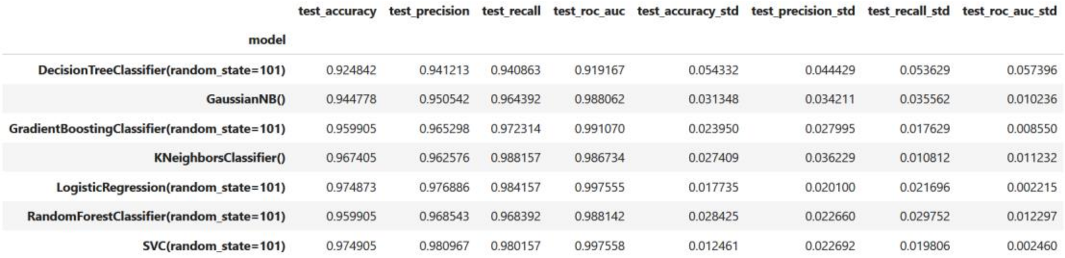
Using only the top 10 features slightly increased the performance of logistic regression and KNN. SD values are notably smaller following feature selection (model performance is less affected by the random selection of instances).



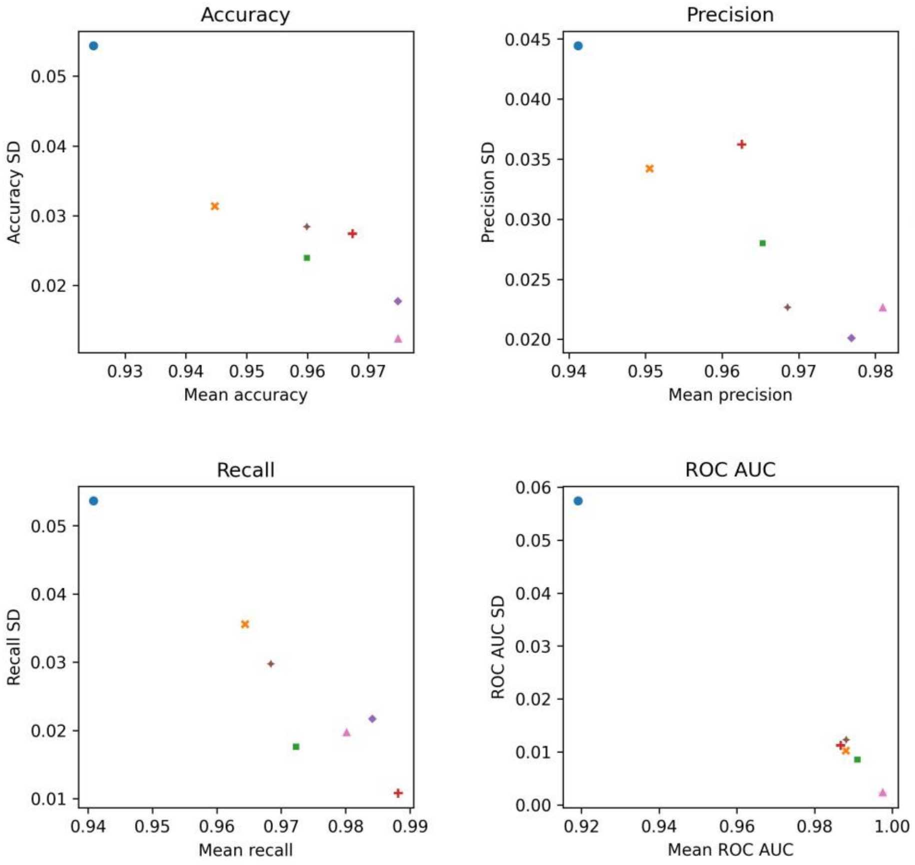
Comparison of different ML models and different sets of features

(Mean and SD values are calculated from the results of cross-validation)

1. All features



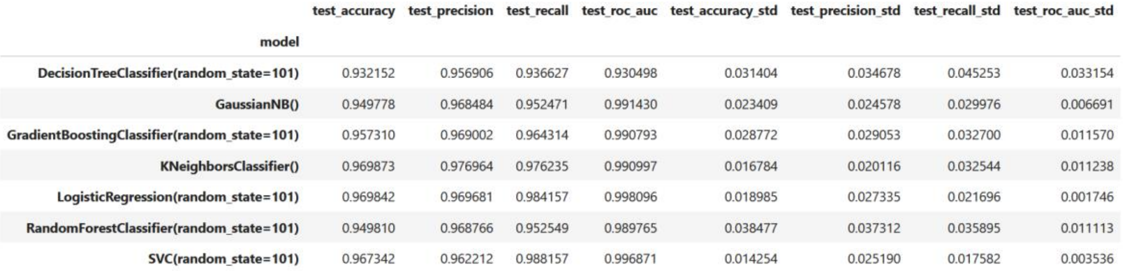
model	test_accuracy	test_precision	test_recall	test_roc_auc	test_accuracy_std	test_precision_std	test_recall_std	test_roc_auc_std
DecisionTreeClassifier(random_state=101)	0.924842	0.941213	0.940863	0.919167	0.054332	0.044429	0.053629	0.057396
GaussianNB()	0.944778	0.950542	0.964392	0.988062	0.031348	0.034211	0.035562	0.010236
GradientBoostingClassifier(random_state=101)	0.959905	0.965298	0.972314	0.991070	0.023950	0.027995	0.017629	0.008550
KNeighborsClassifier()	0.967405	0.962576	0.988157	0.986734	0.027409	0.036229	0.010812	0.011232
LogisticRegression(random_state=101)	0.974873	0.976886	0.984157	0.997555	0.017735	0.020100	0.021696	0.002215
RandomForestClassifier(random_state=101)	0.959905	0.968543	0.968392	0.988142	0.028425	0.022660	0.029752	0.012297
SVC(random_state=101)	0.974905	0.980967	0.980157	0.997558	0.012461	0.022692	0.019806	0.002460



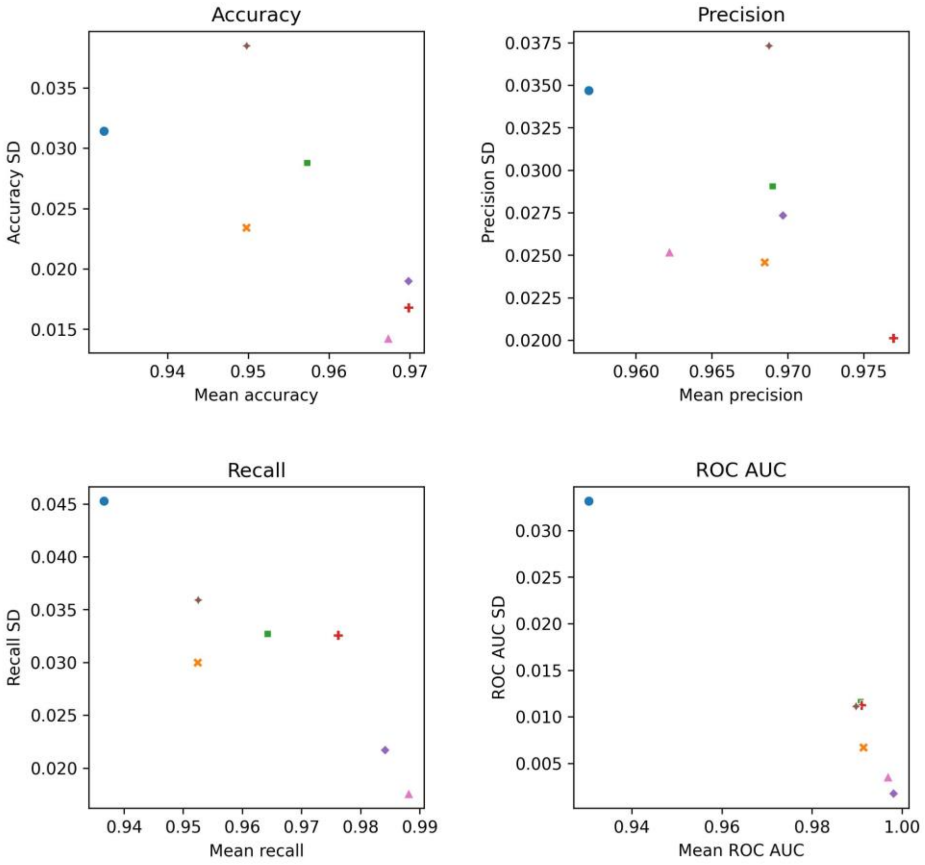
● DecisionTreeClassifier(random_state=101)
× GaussianNB()
■ GradientBoostingClassifier(random_state=101)
+ KNeighborsClassifier()
◆ LogisticRegression(random_state=101)
+ RandomForestClassifier(random_state=101)
▲ SVC(random_state=101)

With a selection of 6 features from the *worst* metrics, model performances showed only slight changes: in most cases a small decrease or no notable change.

3. Selection of only *worst* features



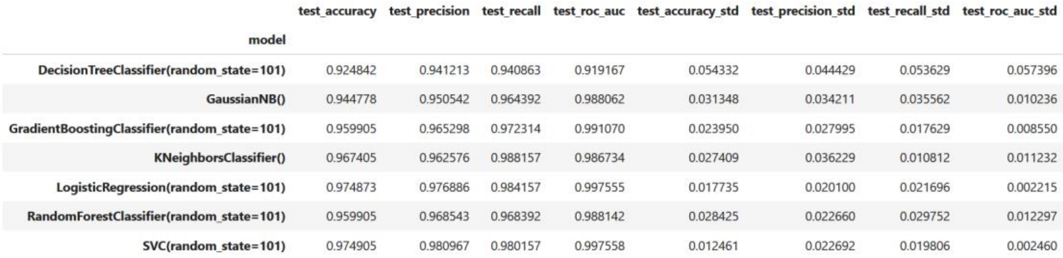
model	test_accuracy	test_precision	test_recall	test_roc_auc	test_accuracy_std	test_precision_std	test_recall_std	test_roc_auc_std
DecisionTreeClassifier(random_state=101)	0.932152	0.956906	0.936627	0.930498	0.031404	0.034678	0.045253	0.033154
GaussianNB()	0.949778	0.968484	0.952471	0.991430	0.023409	0.024578	0.029976	0.006691
GradientBoostingClassifier(random_state=101)	0.957310	0.969002	0.964314	0.990793	0.028772	0.029053	0.032700	0.011570
KNeighborsClassifier()	0.969873	0.976964	0.976235	0.990997	0.016784	0.020116	0.032544	0.011238
LogisticRegression(random_state=101)	0.969842	0.969681	0.984157	0.998096	0.018985	0.027335	0.021696	0.001746
RandomForestClassifier(random_state=101)	0.949810	0.968766	0.952549	0.989765	0.038477	0.037312	0.035895	0.011113
SVC(random_state=101)	0.967342	0.962212	0.988157	0.996871	0.014254	0.025190	0.017582	0.003536



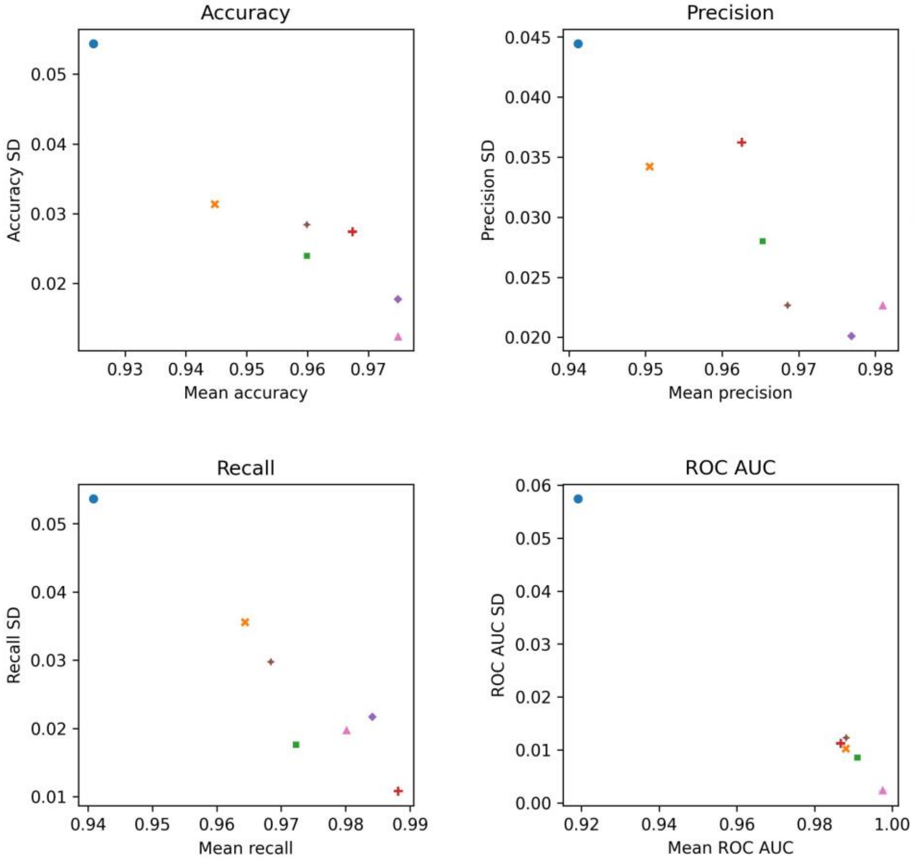
Comparison of different ML models and different sets of features

(Mean and SD values are calculated from the results of cross-validation)

1. All features



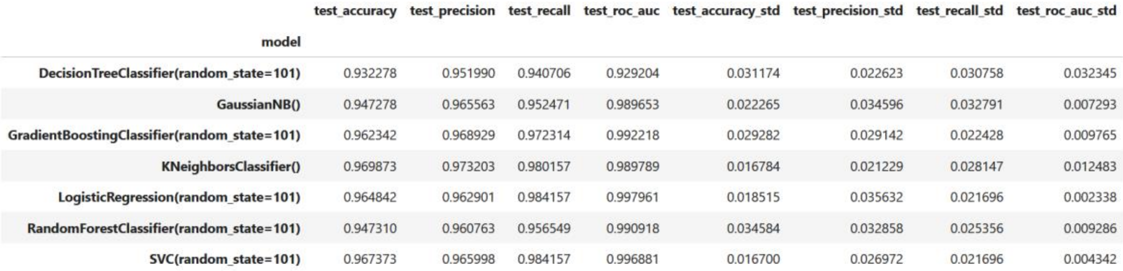
	test_accuracy	test_precision	test_recall	test_roc_auc	test_accuracy_std	test_precision_std	test_recall_std	test_roc_auc_std
model								
DecisionTreeClassifier(random_state=101)	0.924842	0.941213	0.940863	0.919167	0.054332	0.044429	0.053629	0.057396
GaussianNB()	0.944778	0.950542	0.964392	0.988062	0.031348	0.034211	0.035562	0.010236
GradientBoostingClassifier(random_state=101)	0.959905	0.965298	0.972314	0.991070	0.023950	0.027995	0.017629	0.008550
KNeighborsClassifier()	0.967405	0.962576	0.988157	0.986734	0.027409	0.036229	0.010812	0.011232
LogisticRegression(random_state=101)	0.974873	0.976886	0.984157	0.997555	0.017735	0.020100	0.021696	0.002215
RandomForestClassifier(random_state=101)	0.959905	0.968543	0.968392	0.988142	0.028425	0.022660	0.029752	0.012297
SVC(random_state=101)	0.974905	0.980967	0.980157	0.997558	0.012461	0.022692	0.019806	0.002460



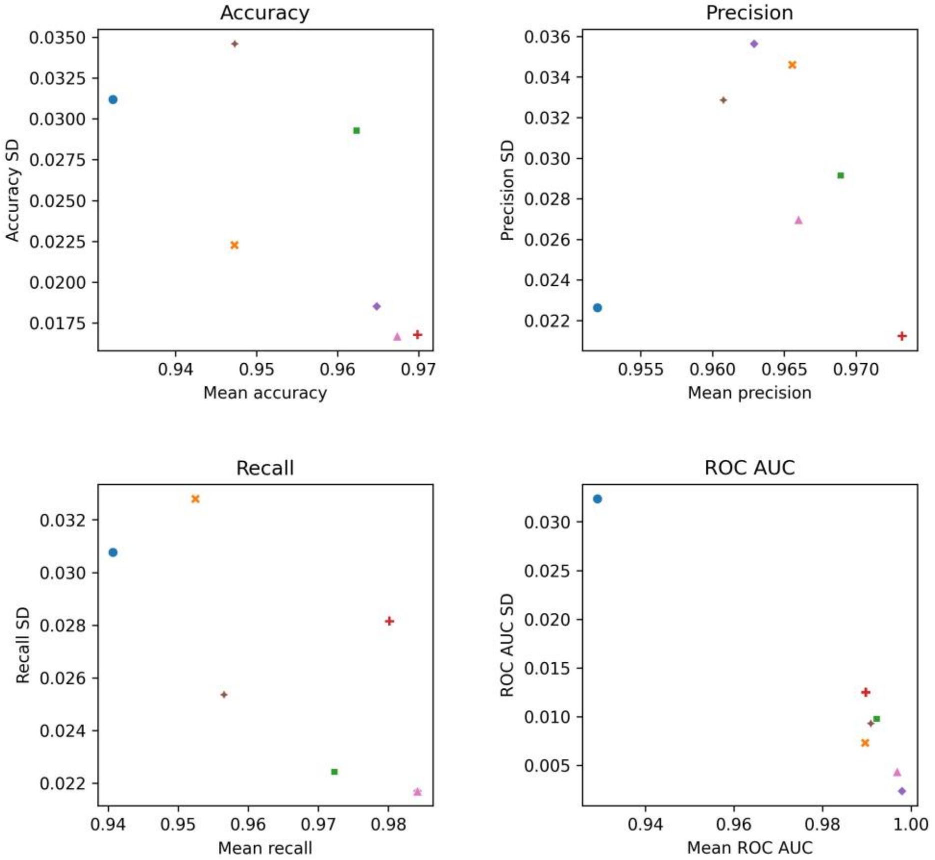
● DecisionTreeClassifier(random_state=101)
× GaussianNB()
■ GradientBoostingClassifier(random_state=101)
+ KNeighborsClassifier()
◆ LogisticRegression(random_state=101)
✦ RandomForestClassifier(random_state=101)
▲ SVC(random_state=101)

The removal of highly correlated features did not notably change model performance compared to selection no. 3. In addition, differences around 0.5-1% between no selection (all features) and selection no. 4 could be negligible. From a practical point of view, only a fraction of the features is able to predict disease outcomes as good as 30 of them.

4. Smallest selection (only 4 features from *worst* metrics)



	test_accuracy	test_precision	test_recall	test_roc_auc	test_accuracy_std	test_precision_std	test_recall_std	test_roc_auc_std
model								
DecisionTreeClassifier(random_state=101)	0.932278	0.951990	0.940706	0.929204	0.031174	0.022623	0.030758	0.032345
GaussianNB()	0.947278	0.965563	0.952471	0.989653	0.022265	0.034596	0.032791	0.007293
GradientBoostingClassifier(random_state=101)	0.962342	0.968929	0.972314	0.992218	0.029282	0.029142	0.022428	0.009765
KNeighborsClassifier()	0.969873	0.973203	0.980157	0.989789	0.016784	0.021229	0.028147	0.012483
LogisticRegression(random_state=101)	0.964842	0.962901	0.984157	0.997961	0.018515	0.035632	0.021696	0.002338
RandomForestClassifier(random_state=101)	0.947310	0.960763	0.956549	0.990918	0.034584	0.032858	0.025356	0.009286
SVC(random_state=101)	0.967373	0.965998	0.984157	0.996881	0.016700	0.026972	0.021696	0.004342


Accuracy
Accuracy SD
Mean accuracy
Precision
Precision SD
Mean precision
Recall
Recall SD
Mean recall
ROC AUC
ROC AUC SD
Mean ROC AUC