Table SI. R package and hyperparameters in R for each model.

Model	R package	Hyperparameters for variable selection	Hyperparameters for prediction
Lasso + StepCox [both]	glmnet, survival	family = "cox", alpha = 1, lambda = cv.fit$lambda.min	direction = both, trace = 0
survival-SVM	survivalsvm		gamma.mu = 1, opt.meth = "ipop"
CoxBoost + survival-SVM	CoxBoost, survivalsvm	start.penalty=500, maxstepno=500	gamma.mu = 1, opt.meth = "ipop"
Ridge	glmnet		family = "cox", alpha = 0, lambda = cv.fit$lambda.min
Lasso + survival-SVM	glmnet, survivalsvm	family = "cox", alpha = 1, lambda = cv.fit$lambda.min	gamma.mu = 1, opt.meth = "ipop"
SuperPC	superpc		type = 'survival', s0.perc = 0.5
CoxBoost + Ridge	CoxBoost, glmnet	start.penalty=500, maxstepno=500	family = "cox", alpha = 0, lambda = cv.fit$lambda.min
Enet [alpha=0.1]	glmnet		family = "cox", alpha = 0.1, lambda = cv.fit$lambda.min
CoxBoost + Enet [alpha=0.1]	CoxBoost, glmnet	start.penalty=500, maxstepno=500	family = "cox", alpha = 0.1, lambda = cv.fit$lambda.min
Enet [alpha=0.2]	glmnet		family = "cox", alpha = 0.2, lambda = cv.fit$lambda.min
Enet [alpha=0.3]	glmnet		family = "cox", alpha = 0.3, lambda = cv.fit$lambda.min
CoxBoost + Enet [alpha=0.3]	CoxBoost, glmnet	start.penalty=500, maxstepno=500	family = "cox", alpha = 0.3, lambda = cv.fit$lambda.min
CoxBoost + Enet [alpha=0.2]	CoxBoost, glmnet	start.penalty=500, maxstepno=500	family = "cox", alpha = 0.2, lambda = cv.fit$lambda.min
Enet [alpha=0.4]	glmnet		family = "cox", alpha = 0.4, lambda = cv.fit$lambda.min
CoxBoost + Enet [alpha=0.4]	CoxBoost, glmnet	start.penalty=500, maxstepno=500	family = "cox", alpha = 0.4, lambda = cv.fit$lambda.min
Lasso + CoxBoost	glmnet, CoxBoost	family = "cox", alpha = 1, lambda = cv.fit$lambda.min	start.penalty=500, maxstepno=500
Enet [alpha=0.5]	glmnet		family = "cox", alpha = 0.5, lambda = cv.fit$lambda.min

CoxBoost	CoxBoost		start.penalty=500, maxstepno=500
CoxBoost + Enet [alpha=0.5]	CoxBoost, glmnet	start.penalty=500, maxstepno=500	family = "cox", alpha = 0.5, lambda = cv.fit$lambda.min
Enet [alpha=0.6]	glmnet		family = "cox", alpha = 0.6, lambda = cv.fit$lambda.min
CoxBoost + Enet [alpha=0.6]	CoxBoost, glmnet	start.penalty=500, maxstepno=500	family = "cox", alpha = 0.6, lambda = cv.fit$lambda.min
CoxBoost + Enet [alpha=0.7]	CoxBoost, glmnet	start.penalty=500, maxstepno=500	family = "cox", alpha = 0.7, lambda = cv.fit$lambda.min
CoxBoost + Enet [alpha=0.8]	CoxBoost, glmnet	start.penalty=500, maxstepno=500	family = "cox", alpha = 0.8, lambda = cv.fit$lambda.min
Enet [alpha=0.8]	glmnet		family = "cox", alpha = 0.8, lambda = cv.fit$lambda.min
Enet [alpha=0.9]	glmnet		family = "cox", alpha = 0.9, lambda = cv.fit$lambda.min
Lasso	glmnet		family = "cox", alpha = 1, lambda = cv.fit$lambda.min
Enet [alpha=0.7]	glmnet		family = "cox", alpha = 0.7, lambda = cv.fit$lambda.min
CoxBoost + Enet [alpha=0.9]	CoxBoost, glmnet	start.penalty=500, maxstepno=500	family = "cox", alpha = 0.9, lambda = cv.fit$lambda.min
CoxBoost + Lasso	CoxBoost, glmnet	start.penalty=500, maxstepno=500	family = "cox", alpha = 1, lambda = cv.fit$lambda.min
Lasso + plsRcox	glmnet, plsRcox	family = "cox", alpha = 1, lambda = cv.fit$lambda.min	nt=10
CoxBoost + plsRcox	CoxBoost, plsRcox	start.penalty=500, maxstepno=500	nt=10
CoxBoost + StepCox [forward]	CoxBoost, survival	start.penalty=500, maxstepno=500	direction = forward, trace = 0
Lasso + StepCox [forward]	glmnet, survival	family = "cox", alpha = 1, lambda = cv.fit$lambda.min	direction = forward, trace = 0
RSF + survival-SVM	randomForestSRC, survivalsvm	ntree = 1000, nodesize = 2, splitrule = 'logrank'	gamma.mu = 1, opt.meth = "ipop"
CoxBoost + SuperPC	CoxBoost, superpc	start.penalty=500, maxstepno=500	type = 'survival', s0.perc = 0.5
StepCox [forward]	survival		direction = forward, trace = 0

plsRcox	plsRcox		nt=10
RSF + Ridge	randomForestSRC, glmnet	ntree = 1000, nodesize = 2, splitrule = 'logrank'	family = "cox", alpha = 0, lambda = cv.fit$lambda.min
RSF + Enet [alpha=0.1]	randomForestSRC, glmnet	ntree = 1000, nodesize = 2, splitrule = 'logrank'	family = "cox", alpha = 0.1, lambda = cv.fit$lambda.min
Lasso + SuperPC	glmnet, superpc	family = "cox", alpha = 1, lambda = cv.fit$lambda.min	type = 'survival', s0.perc = 0.5
RSF + plsRcox	randomForestSRC, plsRcox	ntree = 1000, nodesize = 2, splitrule = 'logrank'	nt=10
RSF + StepCox [forward]	randomForestSRC, survival	ntree = 1000, nodesize = 2, splitrule = 'logrank'	direction = forward, trace = 0
RSF + Enet [alpha=0.2]	randomForestSRC, glmnet	ntree = 1000, nodesize = 2, splitrule = 'logrank'	family = "cox", alpha = 0.2, lambda = cv.fit$lambda.min
RSF + Enet [alpha=0.3]	randomForestSRC, glmnet	ntree = 1000, nodesize = 2, splitrule = 'logrank'	family = "cox", alpha = 0.3, lambda = cv.fit$lambda.min
RSF + Enet [alpha=0.6]	randomForestSRC, glmnet	ntree = 1000, nodesize = 2, splitrule = 'logrank'	family = "cox", alpha = 0.6, lambda = cv.fit$lambda.min
RSF + Lasso	randomForestSRC, glmnet	ntree = 1000, nodesize = 2, splitrule = 'logrank'	family = "cox", alpha = 1, lambda = cv.fit$lambda.min
RSF + Enet [alpha=0.7]	randomForestSRC, glmnet	ntree = 1000, nodesize = 2, splitrule = 'logrank'	family = "cox", alpha = 0.7, lambda = cv.fit$lambda.min
RSF + Enet [alpha=0.5]	randomForestSRC, glmnet	ntree = 1000, nodesize = 2, splitrule = 'logrank'	family = "cox", alpha = 0.5, lambda = cv.fit$lambda.min
RSF + CoxBoost	randomForestSRC, CoxBoost	ntree = 1000, nodesize = 2, splitrule = 'logrank'	start.penalty=500, maxstepno=500
RSF + Enet [alpha=0.9]	randomForestSRC, glmnet	ntree = 1000, nodesize = 2, splitrule = 'logrank'	family = "cox", alpha = 0.9, lambda = cv.fit$lambda.min
RSF + Enet [alpha=0.4]	randomForestSRC, glmnet	ntree = 1000, nodesize = 2, splitrule = 'logrank'	family = "cox", alpha = 0.4, lambda = cv.fit$lambda.min
RSF + Enet [alpha=0.8]	randomForestSRC, glmnet	ntree = 1000, nodesize = 2, splitrule = 'logrank'	family = "cox", alpha = 0.8, lambda = cv.fit$lambda.min
RSF + StepCox [both]	randomForestSRC, survival	ntree = 1000, nodesize = 2, splitrule = 'logrank'	direction = both, trace = 0
RSF + StepCox [backward]	randomForestSRC, survival	ntree = 1000, nodesize = 2, splitrule = 'logrank'	direction = backward, trace = 0

StepCox [both] + Ridge	survival, glmnet	direction = both, trace = 0	family = "cox", alpha = 0, lambda = cv.fit$lambda.min
StepCox [backward] + Ridge	survival, glmnet	direction = backward, trace = 0	family = "cox", alpha = 0, lambda = cv.fit$lambda.min
StepCox [both] + plsRcox	survival, plsRcox	direction = both, trace = 0	nt=10
StepCox [backward] + plsRcox	survival, plsRcox	direction = backward, trace = 0	nt=10
StepCox [both] + Enet [alpha=0.9]	survival, glmnet	direction = both, trace = 0	family = "cox", alpha = 0.9, lambda = cv.fit$lambda.min
StepCox [backward] + Enet [alpha=0.9]	survival, glmnet	direction = backward, trace = 0	family = "cox", alpha = 0.9, lambda = cv.fit$lambda.min
StepCox [both] + Enet [alpha=0.1]	survival, glmnet	direction = both, trace = 0	family = "cox", alpha = 0.1, lambda = cv.fit$lambda.min
StepCox [backward] + Enet [alpha=0.1]	survival, glmnet	direction = backward, trace = 0	family = "cox", alpha = 0.1, lambda = cv.fit$lambda.min
StepCox [both] + Enet [alpha=0.8]	survival, glmnet	direction = both, trace = 0	family = "cox", alpha = 0.8, lambda = cv.fit$lambda.min
StepCox [backward] + Enet [alpha=0.8]	survival, glmnet	direction = backward, trace = 0	family = "cox", alpha = 0.8, lambda = cv.fit$lambda.min
StepCox [both] + Enet [alpha=0.2]	survival, glmnet	direction = both, trace = 0	family = "cox", alpha = 0.2, lambda = cv.fit$lambda.min
StepCox [backward] + Enet [alpha=0.2]	survival, glmnet	direction = backward, trace = 0	family = "cox", alpha = 0.2, lambda = cv.fit$lambda.min
StepCox [both] + Lasso	survival, glmnet	direction = both, trace = 0	family = "cox", alpha = 1, lambda = cv.fit$lambda.min
StepCox [backward] + Lasso	survival, glmnet	direction = backward, trace = 0	family = "cox", alpha = 1, lambda = cv.fit$lambda.min
StepCox [both] + Enet [alpha=0.6]	survival, glmnet	direction = both, trace = 0	family = "cox", alpha = 0.6, lambda = cv.fit$lambda.min
StepCox [backward] + Enet [alpha=0.6]	survival, glmnet	direction = backward, trace = 0	family = "cox", alpha = 0.6, lambda = cv.fit$lambda.min
CoxBoost + GBM	CoxBoost, gbm	start.penalty=500, maxstepno=500	distribution = 'coxph', n.trees = 10000, interaction.depth = 3, n.minobsinnode = 10, shrinkage = 0.001
StepCox [both] + Enet [alpha=0.7]	survival, glmnet	direction = both, trace = 0	family = "cox", alpha = 0.7, lambda = cv.fit$lambda.min

StepCox [backward] + Enet [alpha=0.7]	survival, glmnet	direction = backward, trace = 0	family = "cox", alpha = 0.7, lambda = cv.fit$lambda.min
Lasso + StepCox [backward]	glmnet, survival	family = "cox", alpha = 1, lambda = cv.fit$lambda.min	direction = backward, trace = 0
StepCox [both]	survival		direction = both, trace = 0
StepCox [backward]	survival		direction = backward, trace = 0
CoxBoost + StepCox [both]	CoxBoost, survival	start.penalty=500, maxstepno=500	direction = both, trace = 0
CoxBoost + StepCox [backward]	CoxBoost, survival	start.penalty=500, maxstepno=500	direction = backward, trace = 0
StepCox [both] + Enet [alpha=0.4]	survival, glmnet	direction = both, trace = 0	family = "cox", alpha = 0.4, lambda = cv.fit$lambda.min
StepCox [backward] + Enet [alpha=0.4]	survival, glmnet	direction = backward, trace = 0	family = "cox", alpha = 0.4, lambda = cv.fit$lambda.min
StepCox [both] + Enet [alpha=0.3]	survival, glmnet	direction = both, trace = 0	family = "cox", alpha = 0.3, lambda = cv.fit$lambda.min
StepCox [backward] + Enet [alpha=0.3]	survival, glmnet	direction = backward, trace = 0	family = "cox", alpha = 0.3, lambda = cv.fit$lambda.min
StepCox [both] + CoxBoost	survival, CoxBoost	direction = both, trace = 0	start.penalty=500, maxstepno=500
StepCox [backward] + CoxBoost	survival, CoxBoost	direction = backward, trace = 0	start.penalty=500, maxstepno=500
StepCox [both] + Enet [alpha=0.5]	survival, glmnet	direction = both, trace = 0	family = "cox", alpha = 0.5, lambda = cv.fit$lambda.min
StepCox [backward] + Enet [alpha=0.5]	survival, glmnet	direction = backward, trace = 0	family = "cox", alpha = 0.5, lambda = cv.fit$lambda.min
CoxBoost + RSF	CoxBoost, randomForestSRC	start.penalty=500, maxstepno=500	ntree = 1000, nodesize = 2, splitrule = 'logrank'
RSF + SuperPC	randomForestSRC, superpc	ntree = 1000, nodesize = 2, splitrule = 'logrank'	type = 'survival', s0.perc = 0.5
RSF	randomForestSRC		ntree = 1000, nodesize = 2, splitrule = 'logrank'
Lasso + GBM	glmnet, gbm	family = "cox", alpha = 1, lambda = cv.fit$lambda.min	distribution = 'coxph', n.trees = 10000, interaction.depth = 3, n.minobsinnode = 10, shrinkage = 0.001

RSF + GBM	randomForestSRC, gbm	ntree = 1000, nodesize = 2, splitrule = 'logrank'	distribution = 'coxph', n.trees = 10000, interaction.depth = 3, n.minobsinnode = 10, shrinkage = 0.001
GBM	gbm		distribution = 'coxph', n.trees = 10000, interaction.depth = 3, n.minobsinnode = 10, shrinkage = 0.001
StepCox [both] + survival-SVM	survival, survivalsvm	direction = both, trace = 0	gamma.mu = 1, opt.meth = "ipop"
StepCox [backward] + survival-SVM	survival, survivalsvm	direction = backward, trace = 0	gamma.mu = 1, opt.meth = "ipop"
Lasso + RSF	glmnet, randomForestSRC	family = "cox", alpha = 1, lambda = cv.fit$lambda.min	ntree = 1000, nodesize = 2, splitrule = 'logrank'
StepCox [both] + GBM	survival, gbm	direction = both, trace = 0	distribution = 'coxph', n.trees = 10000, interaction.depth = 3, n.minobsinnode = 10, shrinkage = 0.001
StepCox [backward] + GBM	survival, gbm	direction = both, trace = 0	distribution = 'coxph', n.trees = 10000, interaction.depth = 3, n.minobsinnode = 10, shrinkage = 0.001
StepCox [both] + RSF	survival, randomForestSRC	direction = both, trace = 0	ntree = 1000, nodesize = 2, splitrule = 'logrank'
StepCox [backward] + RSF	survival, randomForestSRC	direction = backward, trace = 0	ntree = 1000, nodesize = 2, splitrule = 'logrank'
StepCox [both] + SuperPC	survival, superpc	direction = both, trace = 0	type = 'survival', s0.perc = 0.5
StepCox [backward] + SuperPC	survival, superpc	direction = backward, trace = 0	type = 'survival', s0.perc = 0.5
Lasso + StepCox [both]	glmnet, survival	family = "cox", alpha = 1, lambda = cv.fit$lambda.min	direction = both, trace = 0