I use random forest, extra tree, ada boost, logistic, SVM, gradient boost, decision tree and KNN classifier. And then use evaluation metrics including accuracy, precision at different levels, recall at different levels, F1, area under curve, and precision-recall curves to compare different model.

Among all classifiers, KNN runs the fastest.
For accuracy matric, DT and AB have the highest accuracy classification score and running time, but AB has relatively much higher precision score.

| | auc | f1 | precision | recall | time | parameters |
|---|---|---|---|---|---|---|
| RF | 0.284931 | 0.755054 | 11.664 | 0.174291 | 0.780225 | {'max_features': 'log2', 'min_samples_split': ... |
| DT | 0.377014 | 0.654297 | 0.048637 | 0.417609 | 0.343612 | {'max_features': 'log2', 'min_samples_split': ... |
| KNN | 0.232258 | 0.585416 | 2.57068 | 0.245633 | 0.220264 | {'n_neighbors': 1, 'weights': 'uniform', 'algo... |
| ET | 0.289694 | 0.712008 | 0.886959 | 0.189267 | 0.61723 | {'max_features': 'log2', 'min_samples_split': ... |
| AB | 0.377014 | 0.654297 | 0.118041 | 0.417609 | 0.343612 | {'n_estimators': 1, 'algorithm': 'SAMME.R'} |
| GB | 0.354703 | 0.640238 | 2075.12 | 0.407964 | 0.313754 | {'n_estimators': 100, 'subsample': 0.5, 'learn... |
| LR | 0.136212 | 0.536573 | 2.57946 | 0.0752159 | 0.720509 | {'penalty': 'l1', 'C': 1} |

For precision metric, RF does the best

| | auc | f1 | precision | recall | time | parameters |
|---|---|---|---|---|---|---|
| RF | 0.284931 | 0.755054 | 11.664 | 0.174291 | 0.780225 | {'max_features': 'log2', 'min_samples_split': ... |
| DT | 0.377014 | 0.654297 | 0.048637 | 0.417609 | 0.343612 | {'max_features': 'log2', 'min_samples_split': ... |
| KNN | 0.232258 | 0.585416 | 2.57068 | 0.245633 | 0.220264 | {'n_neighbors': 1, 'weights': 'uniform', 'algo... |
| ET | 0.289694 | 0.712008 | 0.886959 | 0.189267 | 0.61723 | {'max_features': 'log2', 'min_samples_split': ... |
| AB | 0.377014 | 0.654297 | 0.118041 | 0.417609 | 0.343612 | {'n_estimators': 1, 'algorithm': 'SAMME.R'} |
| GB | 0.354703 | 0.640238 | 2075.12 | 0.407964 | 0.313754 | {'n_estimators': 100, 'subsample': 0.5, 'learn... |
| LR | 0.136212 | 0.536573 | 2.57946 | 0.0752159 | 0.720509 | {'penalty': 'l1', 'C': 1} |

For recall rate, still RF does the best

| | auc | f1 | precision | recall | time | parameters |
|---|---|---|---|---|---|---|
| RF | 0.273486 | 0.759072 | 4.70586 | 0.163862 | 0.826236 | {'max_features': 'log2', 'min_samples_split': ... |
| DT | 0.127516 | 0.5 | 0.0374269 | 0.0681 | 1 | {'max_features': 'sqrt', 'min_samples_split': ... |
| KNN | 0.238618 | 0.706972 | 104.486 | 0.142239 | 0.740088 | {'n_neighbors': 100, 'weights': 'uniform', 'al... |
| ET | 0.127524 | 0.500036 | 1.70907 | 0.0681045 | 1 | {'max_features': 'sqrt', 'min_samples_split': ... |
| AB | 0.127516 | 0.5 | 96.1193 | 0.0681 | 1 | {'n_estimators': 1000, 'algorithm': 'SAMME.R'} |
| GB | 0.127516 | 0.5 | 1.22423 | 0.0681 | 1 | {'n_estimators': 10, 'subsample': 0.1, 'learni... |
| LR | 0.126974 | 0.500799 | 0.149757 | 0.0682108 | 0.916789 | {'penalty': 'l1', 'C': 1e-05} |

For F1 matric, RF performs better.

| | auc | f1 | precision | recall | time | parameters |
|---|---|---|---|---|---|---|
| RF | 0.273486 | 0.759072 | 4.70586 | 0.163862 | 0.826236 | {'max_features': 'log2', 'min_samples_split': ... |
| DT | 0.127516 | 0.5 | 0.0374269 | 0.0681 | 1 | {'max_features': 'sqrt', 'min_samples_split': ... |
| KNN | 0.238618 | 0.706972 | 104.486 | 0.142239 | 0.740088 | {'n_neighbors': 100, 'weights': 'uniform', 'al... |
| ET | 0.127524 | 0.500036 | 1.70907 | 0.0681045 | 1 | {'max_features': 'sqrt', 'min_samples_split': ... |
| AB | 0.127516 | 0.5 | 96.1193 | 0.0681 | 1 | {'n_estimators': 1000, 'algorithm': 'SAMME.R'} |
| GB | 0.127516 | 0.5 | 1.22423 | 0.0681 | 1 | {'n_estimators': 10, 'subsample': 0.1, 'learni... |

| | auc | f1 | precision | recall | time | parameters |
|---|---|---|---|---|---|---|
| **LR** | 0.126974 | 0.500799 | 0.149757 | 0.0682108 | 0.916789 | {'penalty': 'l1', 'C': 1e-05} |