Michael Hering

CSCI 3202

Assignment 3 Write Up

1a) For dataset a, which is almost linearly separable, Logistic regression, Naive Bayes, SVM, and KNN are all fairly accurate and make less mistakes than the Decision Tree, Random Forest, and ADA Boost classifiers. For dataset B, all of the classifiers are accurate except for the Logistic Regression classifier. Because the data is not linearly separable this classifier fails to accurate classify the data. Again for data C, all of the models are fairly accurate except for the Logistic Regression classifier because the data is not linearly separable. Additionally, the Naive bayes classifier does not perform as well on this data as it did on the other data.

1b) A lot can be learned about the classifiers from this example. For the Logistic Regression classifier, we can see that it only performs well on linear separable data, but fails to classify data that is not linearly separable as it produces a linear decision boundary. Classifiers such as Naive Bayes, SVM, and KNN were all able to classify the different datasets with high accuracy, as they are able too produce non-linear decision boundaries that can adapt to the distribution of the data. Interestingly, the Decision Tree and Random Forest classifiers produced slightly less accurate results on data that was almost linearly separable, but not easily categorized by a decision tree as it can only produce decision boundaries that are linear sections and also either horizontal or vertical. These classifiers made more mistakes when attempting to classify data that an SVM or Logistic Regression classifier would very easily classify.

2a1) See tables on next page

Logistic Regression	
AUROC_scores	
0.9798136645962733	
0.9581320450885669	
0.9516908212560387	
0.8679549114331723	
0.8486312399355878	
0.9066022544283414	
0.856682769726248	
0.9114331723027376	
0.9694041867954911	
0.9225589225589226	
accuracy_score	0.8518711484593837
precision_score	0.8557312252964426
recall_score	0.8381375807822584
AUROC_mean	0.917290398812138
AUROC_std	0.04524830470676804

Random Forest	
AUROC_scores	
0.9572981366459627	
0.963768115942029	
0.9235104669887279	
0.8486312399355878	
0.8904991948470209	
0.8655394524959743	
0.855877616747182	
0.9170692431561998	
0.9436392914653785	
0.9469696969696969	
accuracy_score	0.8419511804721889
precision_score	0.7859683794466403
recall_score	0.8609304222347701
AUROC_mean	0.911280245519376
AUROC_std	0.041103503311027155

ADA Boost	
AUROC_scores	
0.9487577639751552	
0.9066022544283414	
0.895330112721417	
0.8663446054750402	
0.7713365539452496	
0.8454106280193237	
0.8582930756843801	
0.9049919484702094	
0.9082125603864735	
0.9494949494949495	
accuracy_score	0.8358711484593838
precision_score	0.8387351778656125
recall_score	0.81777506310115
AUROC_mean	0.8854774452600539
AUROC_std	0.05028142989256276

KNN	
AUROC_scores	
0.6055900621118011	
0.6988727858293076	
0.748792270531401	
0.7020933977455717	
0.6956521739130436	
0.6602254428341385	
0.7085346215780999	
0.6561996779388083	
0.7584541062801933	
0.6616161616161617	
accuracy_score	0.6400072028811525
precision_score	0.5632411067193676
recall_score	0.6270143895492192
AUROC_mean	0.6896030700378526
AUROC_std	0.04324470140127439

Decision Tree	
AUROC_scores	
0.8059006211180125	
0.8325281803542673	
0.792270531400966	
0.6811594202898551	
0.8945249597423511	
0.8236714975845411	
0.786634460547504	
0.8172302737520128	
0.7801932367149758	
0.7937710437710439	
accuracy_score	0.8019839935974391
precision_score	0.7859683794466402
recall_score	0.7911267679688733
AUROC_mean	0.8007884225275529
AUROC_std	0.05066699850467768

2a2) I selected the Logistic Regression classifier as my best model because it has the highest AUROC mean and a low AUROC standard deviation. This tells me that across folds this classifier has a high AUROC and this score does not vary by large amounts between different folds. In addition, this classifier has a higher precision and recall score than other classifiers.

2ac) A KNN classifier classifies a new datapoint by finding the distance, based on some distance function, between this new datapoint and the k closest datapoints in the training set. It will then take the labels of the k nearest neighbors and use them to predict the label of the new datapoint, based on the majority. The AdaBoost classifier that I implemented is a boosted decision tree. This classifier is an ensemble method which combines many “weak” classifiers, in this case Decision Trees, that are combined into one classifier. As this model is learned we can assign weights to “bad” and “good” guesses so that a new classifier will focus on the areas where a previous classifier performed poorly. This “boosting” technique creates “weak” classifiers that focus on trying to improve the performance in these difficult areas.

2b1) I used the gird_search method for hyperparameter tuning. The process is pretty straightforward. For every combination of hyperparameters we will evaluate the performance of a new model on the data using k-fold cross validation and return the parameters that produced the highest performing model.

2b2) The two hyperparameters that we tune for our SVM are C and Gamma. C is the cost of making an error (misclassification) and this will affect the bias and variance of the classifier. A lower C will result in a higher bias and a lower variance, and vice versa. Gamma is a parameter that controls how linear our decision boundary is. A high gamma value can result in decision boundaries forming around individual points themselves in isolation, where a low gamma value can result in a much more linear decision boundary.

2bc) Yes, I was able to improve on my best model from 2a2 using hyperparameter tuning. My tuned random forest classifier achieved an AUROC of 0.93 which is 2% higher than the Logistic Regression classifier's score of 0.91.

2c1) Performance & confusion matrix of my best model:

K-Folds Accuracy: 0.862

K-Folds Precision: 0.834

K-Folds Recall: 0.864

All data Confusion Matrix:

271	0
0	229

2c2) This model does a great job at determining if someone has good or bad credit. Because the recall is slightly higher than the precision, we are more likely to incorrectly classify someone as having good credit when they actually have bad credit.

2d1) RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini', max_depth=10, max_features='auto', max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, min_samples_leaf=1, min_samples_split=2, min_weight_fraction_leaf=0.0, n_estimators=64, n_jobs=None, oob_score=False, random_state=None, verbose=0, warm_start=False)