Practical Application Assignment 17.1: Comparing Classifiers

Problem Statement:

Based on the various information about marketing campaigns for this Portuguese banking institution, we want to build a model to predict whether or not a marketing campaign will lead to a bank term deposit. In order to do this, we will build and assess various models and choose the best one. We will use both accuracy and recall as metrics. Accuracy is most important because in this case there is no outsized harm for false negatives or false positives, so it's better to be generally accurate. However, recall is a good additional metric because for a marketing campaign, we want a model that is able to find all of the positive cases, because a false positive is much better than a false negative so we can actually make the most attempts at customers who will convert.

Findings:

Models with default settings, using accuracy:

Model	Train Time	Train Accuracy	Test Accuracy
Logistic Regression	0.230	0.900	0.902
K Neighbors	0.006	0.912	0.894
Decision Tree	0.160	0.996	0.837
SVM	23.250	0.905	0.901

As can be seen from above, Logistic Regression performed the best in terms of test accuracy with a run time that was not too long. This would be the best model to choose in this case.

Models trained on features selected by sequential feature selection:

Model	Train Time	Train Accuracy	Test Accuracy
Logistic Regression	18.642746	0.899657	0.899328
K Neighbors	18.577167	0.895356	0.895201
Decision Tree	22.741363	0.901287	0.898762
SVM	21.391586	0.899483	0.900138

As can be seen above, this slightly lowered the test accuracy of each model and increased the train time. We would only want to use these models in the case that the number of features needs to be low - for example, due to limited storage.

Models with hyperparameters tuned by Grid Search (had to omit SVM due to extremely long run time):

Model	Train Time	Train Accuracy	Test Accuracy
Logistic Regression	1.930141	0.899691	0.902080
K Neighbors	7.668330	0.913669	0.896091
Decision Tree	0.636424	0.899830	0.900057

I was surprised that this did not do much to improve the test accuracy of the Logistic Regression or KNN models, however it did have a meaningful increase in the Decision Tree model. If we were capitalizing for ease of interpreting the model, I would use this Decision Tree model because of how interpretable decision trees are, and the test score is quite improved from the default.

Models with default setting, using recall:

Model	Train Time	Train Recall	Test Recall
Logistic Regression	0.230	0.233	0.229
K Neighbors	0.006	0.397	0.285
Decision Tree	0.160	0.965	0.338
SVM	23.250	0.267	0.242

Here we can see that Decision Trees vastly outperform the other models in terms of test recall. So if we again are looking to maximize the percentage of true converters that we predict accurately, Decision Trees would be the way to go.

Next Steps:

Given that the hyperparameter-tuned Decision Tree performed almost as well as the highest model (Logistic Regression) in terms of accuracy, and how well it did on the secondary metric (recall), I would recommend that we choose this model. In terms of next steps, we should continue to fine tune the Tree - including pruning it for ease of interpretability, run time, and storage concerns. We can then use this interpretable model to not only make predictions but also train the staff of the marketing team who are making these marketing calls, so that they know which features of a call are most influential to getting the customer to convert and they can prioritize accordingly and even adjust their pitches as needed.

Link to Jupyter Notebook:
https://github.com/AbbyWilson/Model-Comparison/blob/main/prompt_III.ipynb