**Introduction**

Using a data set containing information on 1,389[[1]](#footnote-1) Middlebury College applicants who received on-campus interviews in 2005, we experimented with five machine learning algorithms from Python’s scikit-learn package (four supervised – LinearSVC, KNeighborsClassifier, AdaBoostClassifier with decision stumps, and RandomForestClassifier – and one unsupervised – KMeans) in order to predict both the binary admission problem as well as a multiclass variant that included five groups based on both admission type and enrollment.

**Experimental Setup**

Our data came from an old ECON 0211 (Regression) lab, and contained nearly 1,600 examples. Each example had the following raw features: current state/region of residence, SAT verbal score, SAT math score, and stated major of interest. We added the additional more general categories of “census region” (based on current state/region of residence) and “major area” (based on stated major of interest). Unfortunately, we had to remove many examples as they lacked one or more features, so our final data set size ended up as 1,389.

We conducted two main experiments. In the first, we attempted to use the four supervised learning algorithms in order to predict whether or not a student was admitted (regardless of whether admission was for September of February). In order to evaluate each classifier, we first tweaked certain parameters for each attempting to optimize the accuracy of the same single 10-fold cross validation, and then for final evaluation performed 100 randomly shuffled 10-fold cross validations for each (averaging the 10 individual folds separately over all iterations) and then performed a paired t-test between each of these results. This t-test matrix allowed us to determine whether or not the deviation in overall accuracy (the mean of the 10 folds for any single classifier) was the result of randomness or actual differences between classifiers.

In the second experiment, we engineered a new label for each example that divided the data set into five unique classes: not admitted, admitted in September but didn’t enroll, admitted in September and did enroll, admitted in February but didn’t enroll, and admitted in February and did enroll. We then trained a KMeans clustering algorithm (with its default parameters) and then had it return the size of each cluster. In order to get a rough approximation of accuracy, as there was no easy way to obtain the examples associated with each cluster label (and the 0-4 labels for our five classes did not match up with the 0-4 labels for each cluster) we compared the distribution of cluster labels to that of example labels.

**Results**

For our first experiment, in order of best to worst accuracy in predicting admission, the optimized supervised learning algorithms ranked: LinearSVC (~67.5%), AdaBoostClassifier (~66.4%), RandomForestClassifier (~66.2%), and KNeighborsClassifier (~65.4%). The t-test matrix comparing the results of the 100-time average of random 10-fold cross validations is on the following page.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | AdaBoostClassifier | KNeighborsClassifier | LinearSVC | RandomForestClassifier |
| AdaBoostClassifier | X | X | X | X |
| KNeighborsClassifier | 0.001006785 | X | X | X |
| LinearSVC | 0.000835357 | 4.61097E-07 | X | X |
| RandomForestClassifier | 0.214162703 | 0.000235304 | 3.90891E-05 | X |

As you can see, the resulting *p* from each t-test is significant with the exception of that between RandomForestClassifier and AdaBoostClassifier (although this is to be expected as our AdaBoostClassifier used decision stumps and a RandomForestClassifier also involves multiple decision trees). This suggests that the differences in accuracy between all other classifiers are due to the nature of the classifiers themselves, and so comparing accuracy is sufficient (with the exception of comparing AdaBoostClassifier and RandomForestClassifier). But as LinearSVC has a higher accuracy than all other classifiers (and its accuracy is statistically significant), we can conclude that it performs better on this data than any other classifier we tested does.

For our second experiment, the evaluation technique was unfortunately not nearly as technical as the KMeans algorithm in scikit-learn didn’t have an easy way to obtain which examples are associated with each cluster. As a result, each of its clusters – labeled 0-4 – did not match up with our labels – also 0-4. In order to get a rough approximation of whether or not the clusters were reasonably accurate, aside from the “score” function which returns the negation of the value of the k-means objective, we compared the distribution of cluster labels and that of the actual labels (see below).

The k-means objective value of 4896079.60594 is *significantly* high (and minimizing it with zero being perfect is the goal). In order to confirm that this data is not well suited to the clustering task we performed, we looked at label distribution. As the cluster label distribution doesn’t come close to matching that of the labels, we can safely conclude (especially given the high score value) that KMeans did not perform well at all.

**Conclusion**

As a result of our experiments, we can safely conclude that:

* The lack of features in our data likely resulted in the mediocre accuracy values (only slightly better than guessing the majority label – ~61.6%)
  + More examples would have also probably helped
* The Admissions Office is busy with ED1 (didn’t get us new data in time)
* LinearSVC performs best of the supervised algorithms we tried
* KMeans did not perform well given our chosen multiclass labels

1. The original data set had more examples, but we opted to remove any examples missing *any* feature values (given the small number of features) as well as all examples for which admitted=0 and enrolled=1 (we assume these students were admitted under special circumstances and are therefore outliers). [↑](#footnote-ref-1)