Chris Brown and Colby Wight
Data Science Incubator CS 5830/6830
Presentation Slides

Project 4 - KNN

Introduction

The first part of the project is focused on learning how to use and create the k-nearest-neighbor classification algorithm. The dataset we used was the Cleveland Heart Disease dataset. It includes a few hundred rows and about a dozen attributes. It has both categorical and continuous data in addition to invalid values. As a result, we had to preprocess the data by standardizing all the continuous attributes, one-hot encode the categorical attributes, binarize the target class, and drop rows that had invalid attributes when those attributes were ones used in our algorithm. By batching dropping sets of attributes and measuring the F1 score of the remaining attributes, we were able to find optimal hyperparameters given K.

Methods

Our first approach to finding the optimal K value was to simply use every attribute, after the preprocessing steps described in the introduction, and evaluate every K in the range from [1, 20]. We found that there was a 'bell curve' over K=6 which meant that the best K values were usually [3, 5, 7, 9]. We used all four of these values in our tests for attributes.

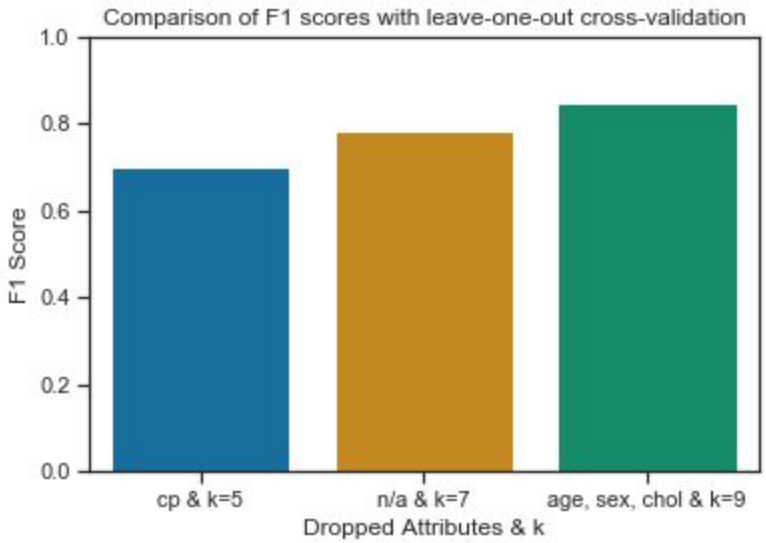
To find the optimal attributes, we batch what we thought were related attributes in 5 groups. With one group at a time, we made every permutation of the group. We iteratively dropped each permutation and computed the F1 score to see which permutation gave the best F1 score. Once we finished a group, we would drop the attributes that boost the F1 score the most, then go onto the next group. Repeating this process five times gave us some respectable results over the baseline of dropping no attributes.

Our final method was a brute force approach by trying every possible permutation of attributes with the four desirable K values. We got our best results here, but as it's not an interesting approach and it may be hugely overfitted, we chose not to use the attribute set that produced the best F1 score.

Results

Our best results were all around an F1 score of 85% when using leave-one-out cross-validation. Our worst results were around an F1 score of 69% when also using leave-one-out cross-validation. This shows there is a clear improvement from our feature selection. The best attributes to drop consistently include age, thalach, chol, one of the encodings for restecg, oldpeak, and sometimes sex. The worst attributes to drop interesting enough also included thalach, but also always have some of the chest pain encodings dropped. The worst

results also have either a K=[3,5,9] while the best have K=[7,9]. Combining this knowledge, we were able to construct a final attribute set that achieved a result of 86% accuracy & 85% F1 using leave-one-out cross-validation and 83% accuracy & 80% F1 with 10-fold cross-validation. Our final attribute set dropped age, resetcg, chol, thalach, oldpeak, and 1 encoding of chest pain.



Part II

Introduction

This part of the project deals with breast cancer. More specifically we want to know whether or not a given cell is malignant or benign by looking at various attributes of the cell. We will show how well we can predict this using a model. If this model does predict a cell as being malignant it is important to know the likelihood of that cell actually being malignant. We also want to know what percent of the cells that actually are malignant that are found.

Dataset

We found the dataset on Kaggle. The dataset was in a comma-separated-value format. The label (benign or malignant) was either an 'M' or a 'B' so we converted the M's to 1 and the B's to 0. The rest of the variables were continuous features and there were no missing values to deal with. The features came from 3-D images of the cell nuclei using fine needle aspirate of the breast mass. 10 types of measurements of the cell were taken: radius, texture, perimeter, area, smoothness, compactness, concavity, concave points, and fractal dimension. Multiple measurements for every type of measurement were made: mean, standard error, and 'worst' (or largest). For example, the radius type of measurement has three features: mean cell radius, worst (or largest) cell radius, and radius measurement standard error for a total of 30 features. The class labels are fairly balanced as well with 37 percent of the cells in the dataset being malignant.

Methods

We used a different approach to choose the attributes than in Part I. We went with a filter-based method which compares each attribute to the label and decides which attributes are most important. We used the ANOVA correlation coefficient which is designed for dealing with numerical input variables and a categorical output. We experimented with how many of the top attributes to choose by testing them on different knn models using a grid search for different k-values from 1 to 20. Once we found the best model parameters, we trained it on all of the training data and tested it on the final test set we had held out. We also used the same grid search approach using all of the attributes and fit the best model to compare the results.

Results

We found the top ten attributes using the ANOVA filter method were the following: radius mean, perimeter mean, area mean, concavity mean, concave points mean, radius worst, perimeter worst, and area worst. Here we were able to see that most of the top attributes were measurement means. The other three top attributes were the measurement maximums or 'worst'. It is interesting to note that none of the top attributes were standard errors of measurement. We also did not see any of the measurements dealing with texture, smoothness, or compactness in the top ten indicating that we may not need to make these measurements to get good predictive results. This could be beneficial to know if acquisition of the other measurements is expensive.

We found the best number of neighbors to look at was 11. This final model with 10 attributes and a K of 11 resulted in a test set F1 score of 95.2%. The comparison model that used all of the attributes and a K of 7 obtained a test set F1 score of 93.7%. Here we see that the model with the top ten attributes not only required less attributes, but it performed better. One explanation of why the accuracy decreased would be that the extra attributes are not very helpful in predicting and that they just add noise. It is important to know that if predictive accuracy is the most important there are other feature selection/feature engineering methods that result in better accuracy. For example one kaggle member used principal component analysis (PCA) to create a features and logistic regression to obtain 98% accuracy. We felt that we should not use PCA as we would lose some of the insight about what attributes are not necessary to achieve good results with K-nn.