CS434 Final Project Report

Nicholas Milford

1 Data preprocessing

Principal component analysis can be optionally used to reduce the demensions of the input set. Interestingly, both the 103 feature dataset and complete feature dataset can retain 99.99% of variance with only five features. As one would expect, reducing the input space this drastically does not help the classifier accuracy very much.

Also attempted was resampling the input space to have an equal number of PK-present and PK-absent examples. I was recommended this by one of my capstone group members who had taken the class before and was using deep learning to solve computer vision problems in our capstone project. This ended up not having a significant positive benefit, as all of the algorithms below either worked well enough without it, or performed worse when this resampling was used.

2 Learning algorithms

2.1 Algorithms/methods explored

Three core classifier algorithms were used: logistic regression, neural network, and decision tree. Later on, ensemble methods were used as an easy way to marginally increase accuracy for the decision tree and logistic regression classifiers, leading to the ensemble logistic regression and random forest classifiers that are used in the final results.

The logistic regression algorithm was explored because it was easiest to implement, needing only to repurpose an earlier assignment. I expected this to perform the worst out of any of the algortihms I would attempt, because it's unlikely the data is seperable. This gave me a baseline accuracy to compare other classifiers to.

The decision tree was explored because I wanted to attempt both a regression-based and non-regression-based algorithm. I didn't know what to expect for performance out of this, but as it was also already implemented from an earlier assignment, I decided to try it.

After trying both of the above and getting inadequate results, I decided to try out the neural network as well. Again, modified from an earlier class assignment meant implementing it was rather easy. The hope was that this would exhibit much better performance than the logistic regression and decision tree classifiers used before.

2.2 Final models

The final models used for the prediction files are the ensemble logistic regression classifieri, the random forest, and the neural network.

Logistic regression ensemble parameters:

- Ensemble size: 11 classifiers
- Learning rate: 0.06
- Bag size: 1000 training examples
- Stop condition: 10000 learning epochs OR gradient magnitude ¡ 0.1
- Initial weight range: [-0.05, 0.05]
- Ensemble method: vote weighted by testing accuracy

Random forest parameters:

- Forest size: 11 decision trees
- Depth limit: 10
- Information gain metric: Gini index
- Feature set size: random 10% of input space
- Ensemble method: vote weighted by validation accuracy

Neural network parameters:

- Batch size: 32
- Structure: 1 hidden layer of 100 nodes, 2 output nodes
- Activation function: RELU
- Dropout rate: 0.2
- Momentum: 0.5
- Weight decay: 0
- Stop condition: More than 5 epochs, and validation accuracy decreases twice in a row

3 Parameter Tuning and Model Selection

3.1 Parameter Tuning

For the logistic regression classifier, both the learning rate and stop condition were parameters that had to be tuned. Determining the learning rate was a trial-and-error process to find what would give the fastest gradient descent while still being stable. The stop condition was originally dependent on the magnitude of the gradient, but this led to large variations in how long a classifier took to

train. The logistic regression training process now stops after a fixed number of batches.

The decision tree / random forest parameters were also limited by computing power. With 100 features at minimum and a training set of about 28,000 examples, it takes an unreasonable amount of time to grow the tree to perfect training performance, so pruning isn't feasible. The original plan was to grow the tree to different depths, and use a hold-out validation set to determine the ideal depth, but that didn't end up working very well for the reasons listed above.

The neural network worked great given the parameters that were left there from the last time I used it, giving an AUC of more than 0.9 on the first try. These parameters can definitely be improved with further research, but the optimal settings will be left as an exercise to the reader.

3.2 Model selection

Given that there are three submissions, and three main models, no real model selection had to be done for this project. As stated before, a hold-out validation set is used for some training processes. The hold-out validation set is constructed such that the proportion of PK-present to PK-absent examples matches that of the training set.

4 Results

Do you have any internal evaluation results you want to report?

Nope