Angus Gibbs and Chris Huber
aag233/cmh333
CS 4780

Machine Learning Final Project Report

Team Name Proof

Our team name was “D.C. Comic Universe (AQUAMAN)” for both competitions.

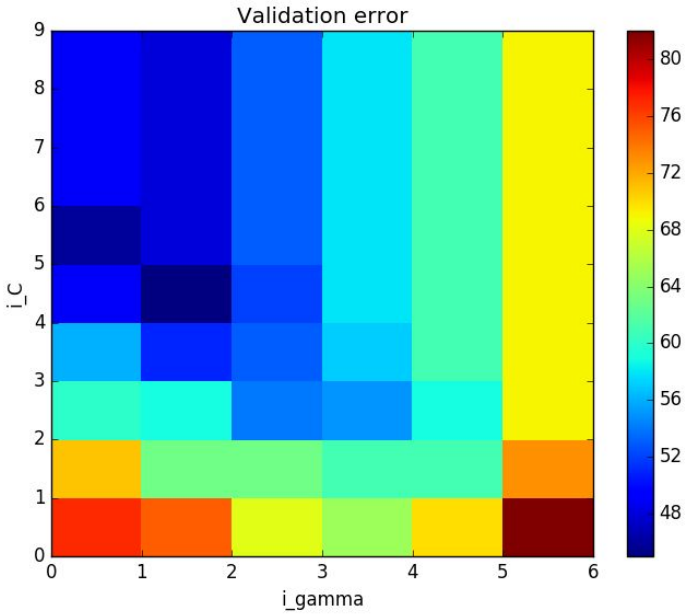
Methods

Preprocessing

First, we used the provided datafile to obtain a dictionary of features created from the image training set. These features are the second to last layer of a ResNet152 network pretrained in ImageNet, and were provided by the course staff. We labeled each of the celebrities with integer values. Then we set up two maps to help us access and organize this data, one to map the labels to celebrity names, and another to map celebrity names to labels.

Initial Attempt

Originally, we tried a multiclass RBF-kernelized SVM (in scikit, it’s called SVC()). The default hyperparameters performed decently producing around around 0.219 validation error. To improve this, we implemented a grid search to find more optimal hyperparameters. There were two parameters to optimize, lamba, the regularization constant, and gamma, which helps modulate the radius of each RBF. We adapted the gridsearch code from project 6. The results of the gridsearch are included below:



For the gridsearch to work, you'll notice, we needed some way of approximating validation error. To do this, we took a 100 sample subset of images from the validation set, and manually labelled them. We used this smaller subset as our gridsearch "validation" data, which allowed us to approximately optimize the parameters. The dark blue in the figure above indicates lower validation error. From this visualization, we clearly identified values 4 and 2^{-9} for lambda and gamma, respectively. Upon passing these more optimal values in for the hyperparameters, we achieved a validation error of about 0.552.

Final Classifier

Our classifier was a linear SVM from the scikit-learn package. In order to extend SVM to handle more than two labels, scikit-learn uses a one-vs-many approach, where, given L labels, it constructs L classifiers. Each classifier $c_i,\ i \in \{0,\ ...,\ L-1\}$ does the binary task of classifying a data vector as either having label i or not label i. Each of these classifiers is run on all of the points to be classified, and the label chosen is the one with the highest positive output of its respective classifier (where "positive" means "classified as label i instead of *not* label i"). We changed the SVM penalty, C, to 32, from its default value of 1.

Sources

We used the scikit-learn and numpy Python packages. We based our grid search off the work that we did in Project 6 for this class.