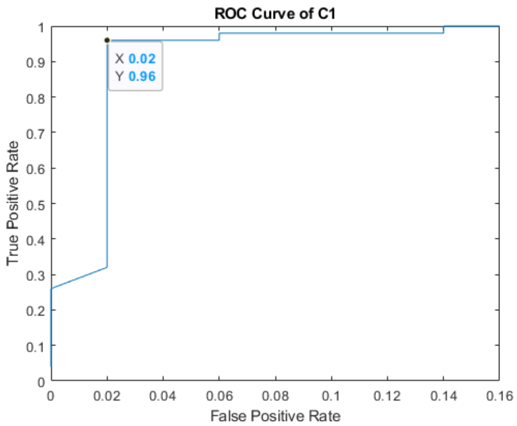
BE5B33KUI: Cybernetics and Artificial Intelligence **(Due: 31/05/20)**

Submission for Assignment 5: Written Report

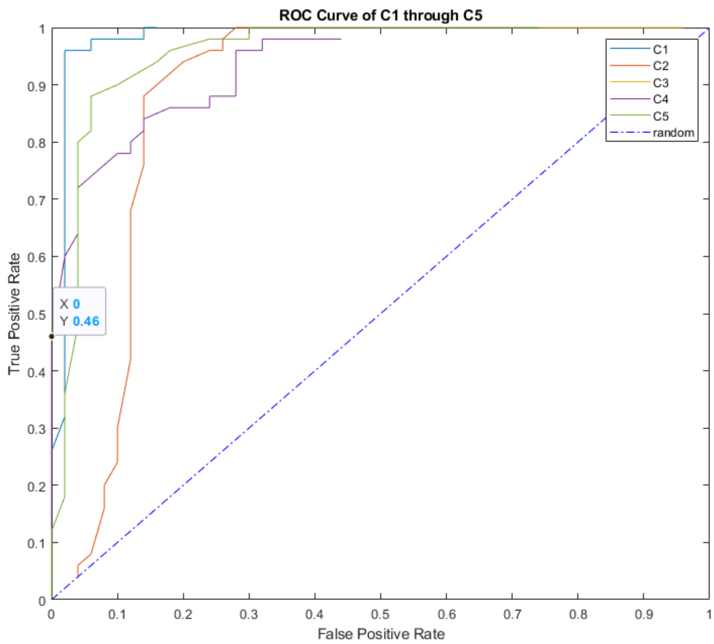
Instructor: Matej Hoffmann *Name:* Ozan Şan, *Netid:* sanozan

Problem 1: Selection of appropriate parameter



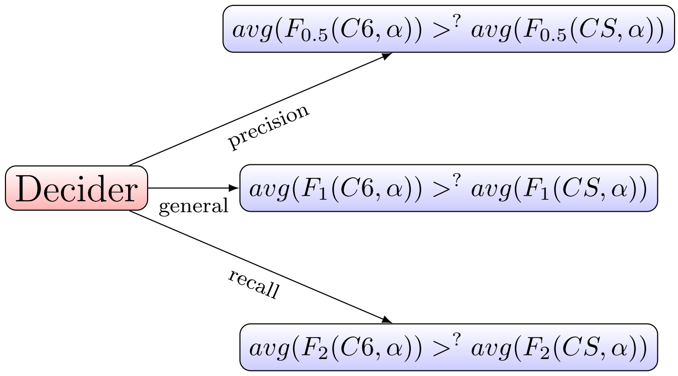
In general, we need to minimize our errors and maximize our true outputs. Deducing from this ROC curve for different parameters α_i for classifier C1, we can see that an ideal parameter would have given us the point $(0, 1)$, meaning maximum sensitivity (TPR) and minimum false-positive rate (FPR) but none of the parameters α_i satisfy this fully. So, we can select the closest point to the desired, which is $(0.02, 0.96)$. This is achieved with values α_i such that $i \in \{23, 24, 25\}$. We can then select $i = 23$ for example, as the distinction between these values do not really matter. Another task could have required a maximum rate of detection (e.g.: Cancer tests) and this would have affected the choice of the parameter, but this is the most general selection of parameter we can come up with.

Problem 2: Top Secret!



In this specific example, we have a situation in our hands that a false positive may be catastrophic, so we must try to observe a minimum of false positives at all costs. In this sense, even though the classifier C4 is not ideal in general situations (area under the curve is one of the lowest), for this specific task, we must use it, because only by using C4 we can maximize TPR with false positives still nonexistent. The marked point represents the most we can push up True Positives without getting a false positive $(FPR = 0, TPR = 0.46)$, and this is achieved by variable α_{12}.

Problem 3: Safety First



For this part, we will consider a metric called F-score. F-score is generally attributed as a "harmonic mean" between precision and recall, possibly with added coefficients (when $\beta \neq 1$). It is modifiable with a parameter β, to give more weight to precision or recall whenever we see fit. To simplify things, our function will take a label l, such that $l \in \{$"recall","general","precision"$\}$, and infer a value for β from the label, as 0.5, 1 or 2 respectively. After we compute confusion matrices for each variable α_i, we can average the F-scores and determine whether the new model is better for the specific application, or not.

```
function decision = decider(ground_truth, selected_classifier, new_classifier, priority)
% we will assume a function computing getConfusionMatrix
% this is also presented in the .zip file as "getConfusionMatrix.m"
% Precision = TP / (TP+FP)
% Recall    = TP / (TP+FN)
% F_beta = (1 + Beta^2) * (Precision * Recall)
%                         ------------------------
%                        (Beta^2 * Precision) + Recall


% keep in mind that we have selected C4 as the best for our specific
% use case in the previous part. We can assign selected_classifier = C4 if need be.

baseline_fscore = 0;
new_fscore = 0;
if priority == "general"
   beta = 1.0;
elseif priority == "precision"
   beta = 0.5;
elseif priority == "recall"
   beta = 2.0;
end


for i=1:size(new_classifier,2)
    conf_new = getConfusionMatrix(GT, new_classifier(:, i));
    precision = conf_new.tp / (conf_new.tp + conf_new.fp);
    recall = conf_new.tp / (conf_new.tp + conf_new.fn);
    fbeta = (1 + beta*beta) * (precision * recall) / (beta*beta*precision) + recall;
    new_fscore = new_fscore + fbeta;
end

for i=1:size(selected_classifier, 2)
    conf_baseline = getConfusionMatrix(GT, selected_classifier(:, i));
    precision = conf_baseline.tp / (conf_baseline.tp + conf_baseline.fp);
    recall = conf_baseline.tp / (conf_baseline.tp + conf_baseline.fn);
    fbeta = (1 + beta*beta) * (precision * recall) / (beta*beta*precision) + recall;
    baseline_fscore = baseline_fscore + fbeta;
end
% Averaging out (on all possible alpha values)
new_fscore = new_fscore / size(new_classifier, 2);
baseline_fscore = baseline_fscore / size(selected_classifier, 2);

decision = (new_fscore > baseline_fscore);
end
```