Appendix

A. Proofs for AUC$_\mu$

A.1. Properties of AUC$_\mu$

Here we show that AUC$_\mu$ satisfies the properties we listed in Section 1.

Theorem A1. *AUC$_\mu$ has the following properties:*

> ***Property 1.*** *If a model gives the correct label the highest probability on every example, then AUC$_\mu$ = 1*
>
> ***Property 2.*** *Random guessing on examples yields AUC$_\mu$ = 0.5*
>
> ***Property 3.*** *AUC$_\mu$ is insensitive to class skew*

Proof. Property 1. Let $\mathcal{M}$ be a multi-class classification model for a task with K classes and let n be the number of examples in the test set. Property 1 assumes that for all n instances $\mathcal{M}$ assigns the correct label the highest probability. The formula for AUC$_\mu$ is an average over separability functions between pairs of classes. Consider the separability function $S(i,j)$ between classes $i < j \leq K$. Let $\hat{\mathbf{p}}^{(a)}$ and $\hat{\mathbf{p}}^{(b)}$ be predictions for instances from classes i and j respectively. Further, let $\mathbf{v}_{i,j}$ be the normal vector to the decision hyperplane between classes i and j derived from the argmax partition matrix $A_{\max}$. As $\hat{\mathbf{p}}^{(a)}$ and $\hat{\mathbf{p}}^{(b)}$ are labeled correctly we have $\tilde{I} \circ O(\mathbf{y}^{(a)}, \mathbf{y}^{(b)}, \hat{\mathbf{p}}^{(a)}, \hat{\mathbf{p}}^{(b)}, \mathbf{v}_{i,j}) = 1$ as the instances are oriented correctly. We have n_i and n_j instances from classes i and j respectively. Then $S(i,j) = 1$ for any pair of classes i and j.

$$S(i,j) = \frac{1}{n_i n_j} \sum_{a \in D^i, b \in D^j} \tilde{I} \circ O(\mathbf{y}^{(a)}, \mathbf{y}^{(b)}, \hat{\mathbf{p}}^{(a)}, \hat{\mathbf{p}}^{(b)}, \mathbf{v}_{i,j}).$$

There are $K(K-1)/2$ choices of unordered pairs of i and j and thus $K(K-1)/2$ choices of $S(i,j)$, each of value 1.

$$\text{AUC}_\mu = \frac{2}{K(K-1)} \sum_{i<j} S(i,j)$$

$$\text{AUC}_\mu = \frac{2}{K(K-1)} \sum_{i<j} 1 = \frac{2}{K(K-1)} \frac{K(K-1)}{2} = 1$$

Therefore, if a model gives the correct label the highest probability on every example, then AUC$_\mu$ = 1.

Property 2: Random guessing yields AUC$_\mu$ $= 0.5$. If a model randomly guesses the prediction for all points, then any two predictions are equally likely to be oriented correctly or incorrectly via Equation 3. Then, for any separability function $S(i,j) = 0.5$ from the modified indicator function, $\tilde{I}$. As AUC$_\mu$ is an unweighted average of separabilities functions over all choices of i and j, the average of all separability functions is 0.5. Therefore, if a model randomly guesses for all points, then AUC$_\mu = 0.5$.

Property 3: AUC$_\mu$ is insensitive to class skew. AUC$_\mu$ is an unweighted average over separability functions. Each separability function, $S(i,j)$, can be computed using the standard two class AUC algorithm by first ranking all instances in classes i and j by using the two-class decision boundary derived from the partition matrix. As AUC is insensitive to class skew, so too are the individual separability functions. The calculation of AUC$_\mu$ is an unweighted average of each $S(i,j)$, and thus changes in class skew will not change the value of AUC$_\mu$. Therefore, AUC$_\mu$ is insensitive to class skew. □

A.2. Partition Matrix

Theorem A2. *Let $\mathcal{M}$ be a model trained to perform a binary classification task. Let A be a 2×2 partition matrix with diagonal zeros and all other entries positive. Then A has no effect on the ranking of predictions from $\mathcal{M}$*

Proof. A typical binary classification model will output a single value, e.g. $\hat{y}^{(a)} = u$, corresponding to the probability of an instance belonging to one of the two classes (typically the positive class). Here we will use an equivalent model output in its categorical distribution form, e.g. $\hat{\mathbf{p}}^{(a)} = [u, 1-u]$. Let $\mathbf{x}^{(a)}$ and $\mathbf{x}^{(b)}$ be two instances with the true labels of these two instances $\mathbf{y}^{(a)} = [1,0]$ and $\mathbf{y}^{(b)} = [0,1]$. Let $\hat{\mathbf{p}}^{(a)} = [u, 1-u]$ and $\hat{\mathbf{p}}^{(b)} = [v, 1-v]$ be the predicted categorical distributions for $\mathbf{x}^{(a)}$ and $\mathbf{x}^{(b)}$ respectively. Recall that in the standard calculation of AUC, if $u > v$ then $\mathbf{x}^{(a)}$ and $\mathbf{x}^{(b)}$ are ranked correctly. We define our partition matrix, A, with $\alpha, \beta > 0$, following the rules presented in Section 3.3.

$$A = \begin{bmatrix} 0 & \alpha \\ \beta & 0 \end{bmatrix}$$

Without loss of generality, let the first class be our positive class and the second class be our negative class. We calculate our orthogonal vector to our hyperplane $\mathbf{v}_{i,j} = [-\beta, \alpha]$. We can now plug in our values into Equation 3 to inspect how our orientation function is influenced by our choice of A. We calculate $(\mathbf{y}^{(a)} - \mathbf{y}^{(b)}) = [1, -1]$ and $(\hat{\mathbf{p}}^{(a)} - \hat{\mathbf{p}}^{(b)}) = [u - v, v - u]$. Further, $\mathbf{v}_{i,j} \cdot (\mathbf{y}^{(a)} - \mathbf{y}^{(b)}) = -(\beta + \alpha)$ and $\mathbf{v}_{i,j} \cdot (\hat{\mathbf{p}}^{(a)} - \hat{\mathbf{p}}^{(b)}) = (\beta + \alpha)(v - u)$. Our final expression yields $(\beta + \alpha)^2(u - v)$. This expression is only positive when $u > v$. Thus, the orientation is solely determined by u and v, and that if $u > v$ our orientation is correct. This is the same result as the standard AUC calculation and we therefore conclude that binary classification is a special case

for the calculation of AUC that does not require a partition matrix. □

Corollary A2.1. *When $K = 2$, AUC$_\mu$ simplifies to the Mann-Whitney U-statistic formulation of AUC.*

Proof. Let $\mathbf{x}^{(a)}$ and $\mathbf{x}^{(b)}$ be two instances with the true labels of these two instances $\mathbf{y}^{(a)} = [1, 0]$ and $\mathbf{y}^{(b)} = [0, 1]$. Let $\hat{\mathbf{p}}^{(a)} = [u, 1-u]$ and $\hat{\mathbf{p}}^{(b)} = [v, 1-v]$ be the predicted categorical distributions for $\mathbf{x}^{(a)}$ and $\mathbf{x}^{(b)}$ respectively. As shown in the proof of Theorem 5.1, $\hat{\mathbf{p}}^{(a)}$ and $\hat{\mathbf{p}}^{(b)}$ will be ranked correctly only if $u > v$. This is exactly the test performed in Equation 1, the Mann-Whitney U-statistic version of AUC. Therefore, when $K = 2$, AUC$_\mu$ is equivalent to AUC. □

Theorem A3. *Let $\mathcal{M}$ be a model trained for a multi-class classification task with $K > 2$ classes. Then the ranking of predictions from $\mathcal{M}$ is not independent of the choice of $K \times K$ partition matrix, hence calculating AUC$_\mu$ requires a partition matrix.*

Proof. Assume that the AUC$_\mu$ for $\mathcal{M}$ is independent of the choice of partition matrix, A, for the task. Then for any two points, $\hat{\mathbf{p}}^{(a)}$ and $\hat{\mathbf{p}}^{(b)}$, the ranking of these points is not changed by A. We show that there is at least one counterexample to our assumption for any $K > 2$. We first show this for $K = 3$ and then describe how this can be easily generalized to choices of $K > 3$.

Let $K = 3$, $\hat{\mathbf{p}}^{(a)} = [.5, .45, .05]$, $\hat{\mathbf{p}}^{(b)} = [.35, .4, .25]$, and $\mathbf{y}^{(a)} = [1, 0, 0]$, $\mathbf{y}^{(b)} = [0, 1, 0]$. Let A and A' be two partition matrices. Let $A_{1,\cdot} = [0, 1, 1]$ and $A_{2,\cdot} = [1, 0, 1]$. We find $\mathbf{v}_{i,j} = [-1, 1, 0]$ and $O(\mathbf{y}^{(a)}, \mathbf{y}^{(b)}, \hat{\mathbf{p}}^{(a)}, \hat{\mathbf{p}}^{(b)}, \mathbf{v}_{i,j}) = 0.2$ showing that our points are oriented correctly. Now let $A'_{1,\cdot} = [0, 4, 1]$ and $A'_{2,\cdot} = [1, 0, 1]$. We find $\mathbf{v'}_{i,j} = [-1, 4, 0]$ and $O(\mathbf{y}^{(a)}, \mathbf{y}^{(b)}, \hat{\mathbf{p}}^{(a)}, \hat{\mathbf{p}}^{(b)}, \mathbf{v'}_{i,j}) = -0.25$ showing that our points are oriented incorrectly. Therefore, for $K = 3$ there exists at least one pair of points whose ranking is dependent on the choice of partition matrix. This can be readily generalized to $K > 3$ by padding 0s to the end of the vectors $\hat{\mathbf{p}}^{(a)}$, $\hat{\mathbf{p}}^{(b)}$, $\mathbf{y}^{(a)}$, and $\mathbf{y}^{(b)}$. □

Theorem A4. *The expectation over all partition matrices, uniformly distributed over $[0, 1]^{K \times K}$, for a task with K classes is equivalent to the argmax partition matrix, $A_{\max}$, where $(A_{\max})_{i,i} = 0 \; \forall i$ and $(A_{\max})_{i,j} = 1 \; \forall i \neq j$.*

Proof. Let $\mathcal{A}$ be the set of all $K \times K$ partition matrices with diagonal elements zero and off-diagonal elements positive in the space $[0, 1]^{K \times K}$. Now let A be a partition matrix drawn randomly according to a uniform distribution, $\mathcal{U}$, from $\mathcal{A}$. Consider two off diagonal elements in A, $A_{i,j}$ and $A_{k,l}$ with $(i, j) \neq (k, l)$. We find that $\mathbb{E}_{\mathcal{U}} A_{i,j} = \mathbb{E}_{\mathcal{U}} A_{k,l}$ as there is exactly one other $A' \in \mathcal{A}$ where $A'_{i,j} = A_{k,l}$, $A'_{k,l} = A_{i,j}$, and all other elements equal. Therefore, if the expectation of any two random off-diagonal elements in A are equal, then the expectations of all off-diagonal elements in A is equal. Let $\mathbb{E}_{\mathcal{U}} A_{i,j} = \sigma$; then A is equal to $\sigma A_{\max}$, where $A_{\max}$ is the argmax partition matrix with uniform misclassification costs. As noted in (O'Brien et al., 2008), $A_{\max}$, induces the same partitioning as A. Because we chose A at random from $\mathcal{A}$, we find that $\mathbb{E}_{\mathcal{U}} A$ is equivalent to the argmax partition matrix with uniform misclassification costs. □

A.3. Time Complexity

Time Complexity of AUC$_\mu$ Using Argmax Partition Matrix $A_{\max}$

$$\text{AUC}_\mu = \frac{2}{K(K-1)} \sum_{i<j} S(i, j)$$

Here we compute the time complexity of AUC$_\mu$ when the argmax partition matrix, $A_{\max}$ is used. Let K be the number of classes and $n = n_1 + \ldots n_K$ be the total number of instances, and the number of instances for each class respectively. Computing AUC$_\mu$ requires $K(K-1)/2$ repetitions of computing a separability function between an unordered pair of classes. We breakup the time complexity calculation into two parts; we first calculate the time complexity of a single separability function $S(i, j)$, and then we use this to calculate the time complexity of AUC$_\mu$.

We calculate the time complexity of $S(i, j)$ by first noting its relationship with AUC. $S(i, j)$ shares an equivalence to AUC as it is the probability that two instances are ranked correctly by first using the two class decision boundary, $\mathbf{v}_{i,j} \cdot \hat{\mathbf{p}} = 0$, to map the categorical predictions to scalar values that can be ranked. As such, the time to compute $S(i, j)$ is the time to perform the mapping and then the time to calculate the AUC. There are $n_i + n_j$ predictions for each $S(i, j)$ computation, and mapping these categorical distributions to scalar values involves a dot product between two vectors of dimension K. This procedure would normally be of complexity $O(K(n_i + n_j))$, however we next show that when using the argmax matrix this can be performed in time $O(n_i + n_j)$. All elements in $A_{\max}$ are 1 except the diagonal entries which are 0. Thus all elements of $\mathbf{v}_{i,j} = (A_{\max})_{i,\cdot} - (A_{\max})_{j,\cdot}$ are 0 except $(\mathbf{v}_{i,j})_i = -1$ and $(\mathbf{v}_{i,j})_j = 1$. Thus we may calculate $\mathbf{v}_{i,j} \cdot \hat{\mathbf{p}} = \hat{\mathbf{p}}_\mathbf{j} - \hat{\mathbf{p}}_\mathbf{i}$, which is a constant time calculation for each of the $n_i + n_j$ instances. We next compute $S(i, j)$ by calculating the AUC of these ranked instances, and using the AUC time complexity result from (Fawcett, 2006). The time complexity of computing AUC with $n_i + n_j$ instances is $O((n_i + n_j) \log(n_i + n_j))$. Thus, time complexity to calculate $S(i, j)$ is $O((n_i + n_j) \log(n_i + n_j))$ as the time complexity of the ranking is dominated by the complexity of calculating the AUC of the ranked points.

The time complexity for AUC$_\mu$ can be determined by summing the time complexities for each of the $K(K-1)/2$ choices of i and j.

$$O(\sum_{i<j}(n_i+n_j)\log(n_i+n_j))$$

We note here that $\log(n_i+n_j) < \log n$ as $n_i+n_j < n$ and we use this relationship so that we may pull this term out of the sum.

$$< O(\log n \sum_{i<j} n_i + n_j)$$

Next we note that for a given n_i it appears in the sum $K-1$ times as it shows up each time it class i is paired with one of the other $K-1$ classes. We therefore exchange the two sums

$$= O(\log n \sum_{i=1}^{K}(K-1)n_i)$$

$$= O(\log n(K-1)n)$$

Therefore, the time complexity of AUC$_\mu$ is $O(Kn\log n)$.

Time Complexity of AUC$_\mu$ With Any Partition Matrix

For the general case of computing AUC$_\mu$ with any partition matrix, A, the time complexity is $O(Kn(K+\log n))$. When computing the time complexity of AUC$_\mu$ using $A_{\max}$, we noted that $A_{\max}$ had the special property that dot products could be performed in constant time. This characteristic is not true for all choices of A and thus in general the ranking of points has complexity $O(K(n_i+n_j))$. Therefore, computing $S(i,j)$ has complexity $O((n_i+n_j)(K+\log(n_i+n_j)))$. Finding the total time complexity of AUC$_\mu$ can be done in the same manner as the computation above, however this time we substitute $K+\log n$ for $K+\log(n_i+n_j)$ and pull the former term out of the sum. Therefore, the time complexity of AUC$_\mu$ for a general choice of A is $O(Kn(K+\log n))$.

Time Complexity of M

The measure, M, proposed by Hand & Till (2001) is claimed to have a time complexity of $O(K^2 n\log n)$ by Fawcett (2006). An additional contribution of our work is to show that the time complexity of this measure is in fact $O(Kn\log n)$. The calculation follows almost identically our approach for computing the complexity of AUC$_\mu$ using the argmax partition matrix. Let K be the number of classes and $n = n_1 + \ldots n_K$ be the total number of instances, and the number of instances for each class respectively.

$$M = \sum_{i<j}\hat{A}(i,j)$$

Here, $\hat{A}(i,j)$ is computed by averaging two AUC calculations each with n_i+n_j instances. Therefore, the time complexity of computing $\hat{A}(i,j)$ is $O((n_i+n_j)\log(n_i+n_j))$. The expressions of M and AUC$_\mu$ differ only in their separability functions $\hat{A}(i,j)$ and $S(i,j)$. As these functions have the same time complexity, the overall algorithms do as well. Therefore, the time complexity of M is $O(Kn\log n)$.

B. Additional Details

Figure 3 uses the following partition matrix for the dashed-line partitioning.

$$\begin{bmatrix} 0 & 4.6 & 0.4 \\ 0.3 & 0 & 8.26 \\ 2 & 4.13 & 0 \end{bmatrix}$$