Algorithm 1 Algorithm for calculating the delta accuracy between two datasets following a dataset shift, utilizing Biased Random Subspaces. The bias is determined by a probability vector derived from feature importances. *Note: the variables* $\boldsymbol{dfvp}_k$*,* $\boldsymbol{mss}_k$*,* $\boldsymbol{v}_k$ $(k \in \{a, b\})$*,* $\boldsymbol{fi}$*, and* $\boldsymbol{featureBias}$ *are vectors (* $\mathbb{R}^m$*) of feature importance scores (one score for each feature)*

Ensure: $\mathcal{D}^a$ and $\mathcal{D}^b$ are datasets with the same number of features (m).

```
function ComputeFeatureBias($\boldsymbol{v}_a, \boldsymbol{v}_b$)
    $\boldsymbol{fi} \leftarrow \boldsymbol{v}_b - \boldsymbol{v}_a$
    $\boldsymbol{fi} \leftarrow$ MinMaxNormalization($\boldsymbol{fi}$)
    return $\frac{\boldsymbol{fi}}{|\boldsymbol{fi}|_1}$                                  // $\boldsymbol{fi}$ normalized by L1 norm
end function

function DeltaACC($\mathcal{D}^a, \mathcal{D}^b$, $method$)
    $nEstimators \leftarrow 500$
    $maxFeatures \leftarrow \sqrt{m}$
    $splitRatio \leftarrow 0.1$
    $model \leftarrow$ ENB($nEstimators, maxFeatures$)
    $\mathcal{D}^{train}, \mathcal{D}^{testID} \leftarrow trainTestSplit(\mathcal{D}^a, splitRatio)$
    $\mathcal{D}^{testOOD} \leftarrow \mathcal{D}^b$
    $classifier \leftarrow model.train(\boldsymbol{X}^{train}, \boldsymbol{y}^{train})$
    $\widehat{\boldsymbol{y}}^{testID} \leftarrow classifier.predict(\boldsymbol{X}^{testID})$
    $\widehat{\boldsymbol{y}}^{testOOD} \leftarrow classifier.predict(\boldsymbol{X}^{testOOD})$
    $parameters_a \leftarrow (\boldsymbol{X}^a, \boldsymbol{y}^a, model)$
    $parameters_b \leftarrow (\boldsymbol{X}^{testOOD}, \widehat{\boldsymbol{y}}^{testOOD}, model)$
    $\boldsymbol{dfvp}_a \leftarrow$ DiffFeatValProba($parameters_a$)
    $\boldsymbol{dfvp}_b \leftarrow$ DiffFeatValProba($parameters_b$)
    $\boldsymbol{mss}_a \leftarrow$ MinimalSuficientSet($parameters_a$)
    $\boldsymbol{mss}_b \leftarrow$ MinimalSuficientSet($parameters_b$)
    if method = ‘dfvp_b’ then
        $\boldsymbol{featureBias} \leftarrow \boldsymbol{dfvp}_b$
    else if method = ‘dfvp_b-a’ then
        $\boldsymbol{featureBias} \leftarrow$ ComputeFeatureBias($\boldsymbol{dfvp}_a, \boldsymbol{dfvp}_b$)
    else if method = ‘ms_b’ then
        $\boldsymbol{featureBias} \leftarrow \boldsymbol{mss}_b$
    else if method = ‘ms_b-a’ then
        $\boldsymbol{featureBias} \leftarrow$ ComputeFeatureBias($\boldsymbol{mss}_a, \boldsymbol{mss}_b$)
    end if
    $modelAdapted \leftarrow ENB(nEstimators, maxFeatures,$
↪                           $biasedSubspaces = True, \boldsymbol{featureBias})$
    $classifierAdapted \leftarrow modelAdapted.train(\boldsymbol{X}^a, \boldsymbol{y}^a)$
    $\widehat{\boldsymbol{y}}^{testOOD} \leftarrow classifierAdapted.predict(\boldsymbol{X}^{testOOD})$
    return $ACC(\widehat{\boldsymbol{y}}^{testOOD}, \boldsymbol{y}^{testOOD}) - ACC(\widehat{\boldsymbol{y}}^{testID}, \boldsymbol{y}^{testID})$
end function
```