1 Proposed Approach

Consider two datasets: $\mathcal{D}^a = \{(\boldsymbol{X}^a, \boldsymbol{y}^b)\}$ and $\mathcal{D}^b = \{\boldsymbol{X}^b\}$, where only $\mathcal{D}^a$ was labeled. Both $\boldsymbol{X}^a$ and $\boldsymbol{X}^b$ represent feature vectors in the same space ($\mathbb{R}^m$) and may have different sizes in terms of the number of instances. From $\mathcal{D}^a$, we trained a model, an ensemble of *naïve* Bayes, to evaluate the importance of attributes in $\boldsymbol{X}^a$. The same model predicts labels for $\boldsymbol{X}^b$, and the predicted labels ($\widehat{\boldsymbol{y}}^b$) are then utilized to evaluate the importance of features in $\mathcal{D}^b$.

For testing purposes, we utilized $\mathcal{D}^b$ with known labels. However, it's important to note that the models did not have access to $\boldsymbol{y}^b$ during the training phase. This information was solely used in the production of the method's quality measure (Δ_{acc}).

The model could provide two feature importance measures. They were named Difference Between Conditional Probabilities (DBCP) and Minimal Sufficient Set (MSSF) [**?**], and were employed to compute four distinct strategies:

1. **DBCP**($\mathcal{D}^b$): The DBCP method calculates the feature importances for $\mathcal{D}^b$.

2. **DBCP**($\mathcal{D}^b$) − **DBCP**($\mathcal{D}^a$): This strategy calculates the difference in DBCP feature importances between $\mathcal{D}^b$ and $\mathcal{D}^a$.

3. **MSS**($\mathcal{D}^b$): The MSS method calculates the feature importances for $\mathcal{D}^b$.

4. **MSS**($\mathcal{D}^b$) − **MSS**($\mathcal{D}^a$): This strategy calculates the difference in MSS feature importances between $\mathcal{D}^b$ and $\mathcal{D}^a$.

The result of each strategy was min-max normalized to ensure values are between 0 and 1. The normalized values are then transformed into probability distributions by dividing each element by the sum of the absolute values of the same vector, following the L1 normalization. Each resulting probability distribution ($\boldsymbol{\Lambda}_k$), where k represents a specific strategy, satisfies $\sum_{j=1}^{m} \Lambda_{kj} = 1, \quad \forall k \in \{1, 2, 3, 4\}$. These distributions are then utilized as bias vectors in a Biased Random Subspace model.

Algorithm 1

```
1: function DbcpFromDb($\boldsymbol{X}^a, \boldsymbol{y}^a, \boldsymbol{X}^b$)
2:     ModelA.Train($\boldsymbol{X}^a, \boldsymbol{y}^a$)
3:     $\widehat{\boldsymbol{y}}^b \leftarrow$ ModelA.Predict($\mathcal{X}^b$)
4:     ModelB.Train($\boldsymbol{X}^b, \widehat{\boldsymbol{y}}^b$)
5:     $\boldsymbol{w} \leftarrow$ ModelB.FeatureImportances( )
6:     return $\boldsymbol{w}$
7: end function
```

Algorithm 2

```
1: function DbcpFromDb-DbcpFromDa($\boldsymbol{X}^a, \boldsymbol{y}^a, \boldsymbol{X}^b$)
2:     ModelA.Train($\boldsymbol{X}^a, \boldsymbol{y}^a$)
3:     $\widehat{\boldsymbol{y}}^b \leftarrow$ ModelA.Predict($\mathcal{X}^b$)
4:     ModelB.Train($\boldsymbol{X}^b, \widehat{\boldsymbol{y}}^b$)
5:     $\mathbf{DBCP}^a \leftarrow$ ModelA.FeatureImportances( )
6:     $\mathbf{DBCP}^b \leftarrow$ ModelB.FeatureImportances( )
7:     $\boldsymbol{w} \leftarrow \mathbf{DBCP}^b - \mathbf{DBCP}^a$
8:     return $\boldsymbol{w}$
9: end function
```

Algorithm 3

```
1: function MssFromDb($\boldsymbol{X}^a, \boldsymbol{y}^a, \boldsymbol{X}^b$)
2:     ModelA.Train($\boldsymbol{X}^a, \boldsymbol{y}^a$)
3:     $\boldsymbol{w} \leftarrow$ ModelA.MinimalSufficientSet($\boldsymbol{X}^b$)
4:     return $\boldsymbol{w}$
5: end function
```

Algorithm 4

```
1: function MssFromDb-MssFromDa($\boldsymbol{X}^a, \boldsymbol{y}^a, \boldsymbol{X}^b$)
2:     ModelA.Train($\boldsymbol{X}^a, \boldsymbol{y}^a$)
3:     $\mathbf{MSS}^a \leftarrow$ ModelA.MinimalSufficientSet($\boldsymbol{X}^a$)
4:     $\mathbf{MSS}^b \leftarrow$ ModelA.MinimalSufficientSet($\boldsymbol{X}^b$)
5:     $\boldsymbol{w} \leftarrow \mathbf{MSS}^b - \mathbf{MSS}^a$
6:     return $\boldsymbol{w}$
7: end function
```

Algorithm 5

```
1: function ModelAdaptation_BRS($\boldsymbol{X}^a, \boldsymbol{y}^a, \boldsymbol{X}^b$, FeatImp())
2:     $\boldsymbol{w} \leftarrow$ FeatImp($\boldsymbol{X}^a, \boldsymbol{y}^a, \boldsymbol{X}^b$)
3:     $\boldsymbol{v} \leftarrow$ MinMaxNormalization($\boldsymbol{w}$)                    //
4:     $\boldsymbol{p} \leftarrow \frac{\boldsymbol{v}}{\|\boldsymbol{v}\|_1}$                  // return a probability vector
5:     AdaptedModel() $\leftarrow$ TrainAdaptedModel($\boldsymbol{X}^a, \boldsymbol{y}^a, \boldsymbol{p}$)
6:     return AdaptedModel()
7: end function
```