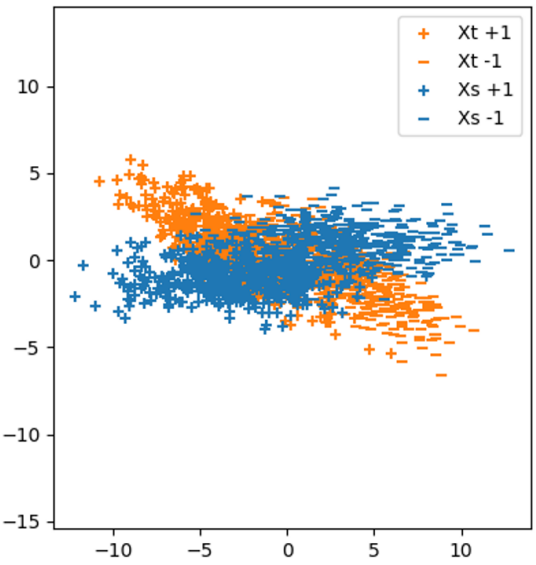
1. In this problem, you will test the subspace alignment (SA) algorithm against supervised learning (SL) methods on 2D synthetic data.

   Data in the source domain $\mathcal{X}_s$ is a mixture of Gaussian:

   $$P_S = P_S(x|y=1)P_S(y=1) + P_S(x|y=-1)P_S(y=-1)$$

   $P_S(x|y=1)$ is Gaussian with mean $\begin{bmatrix}-3\\-1\end{bmatrix}$ and covariance $\begin{bmatrix}10 & 0\\0 & 1\end{bmatrix}$. $P_S(x|y=-1)$ is Gaussian with mean $\begin{bmatrix}3\\1\end{bmatrix}$ and covariance $\begin{bmatrix}10 & 0\\0 & 1\end{bmatrix}$. $P_S(y=1) = P_S(y=-1) = 0.5$.

   Data in the target domain $\mathcal{X}_T$ follows the same distribution except that it rotates axes from source domain by θ. An example of the source data and target data with $\theta = 30°$ is visualized as below.



   The sampled source domain data D_S has $N_S = 1000$ points, and the sampled target domain data D_T has $N_T = 1000$ points. The testing data D_{Test} in the target domain has $N_{Test} = 2000$ points. Throughout this problem, we keep all feature dimensions in SA.

   Lots of the functions needed have been provided to you in the assignment material. You will implement the TO-DO parts in the code and answer the questions below. Note that you will always call the *experiment* function to get results for a certain setting, but feel free to add your own sections for plotting and so on. Do not remove or change the random seed setting in that function so that your answers align with our solution.

   You may read through the whole problem before you read the code. Try to get a good understanding of the provided code before implementing your own part.

(a) Use the provided *generate_data* function to draw the datasets for θ from 0° to 180° with a step size of 30°. For each θ, visualize D_S and D_T on the same plot. You may use the same or similar notations as the picture shown above. Include the plots in your answer.

Note: visualize the data before doing standardization on each feature dimension, i.e., in the original feature domain.

(b) The default steps of SA are as follows.

> Input: **standardized** data $\{x_{S,i}, y_{S,i}\}_{i=1}^{N_S}$, $\{x_{T,i}\}_{i=1}^{N_T}$.
>
> Step 1: calculate the PCA for $\mathcal{X}_s$ and $\mathcal{X}_T$, respectively.
>
> Step 2: calculate the transformation matrix M.
>
> Step 3: train a base classifier Q with D_S in the aligned source domain $\mathcal{X}_a$.

(i) Use an LDA classifier as Q. For θ from $0°$ to $180°$ with a step size of $30°$ (the datasets you draw in (a)), do SA training (as above) and also supervised training (in the original source domain, using standardized data) using the same base classifier. Test your trained SA and SL classifiers on D_{Test}. Draw the plot of testing accuracy vs. θ for SA and SL on the same figure.

(ii) Repeat (i) for Q=Linear SVM

(iii) Repeat (i) for Q=kNN with $k = 31$.

(iv) Comment on your results. Which performs better, SA or SL?

Tip for (i)-(iv): your answers might not be what you expected; that doesn't mean your results are wrong.

Hint: Search for TO-DO in the provided code and fill in those parts. Feel free to use the provided plotting function or create your own. Note that SL algorithm should also work on standardized data.

(c) You may have found that, when SA gives testing accuracy lower than 0.5, the classifier learned something but is predicting the labels backwards. We can improve the performance by considering the PCA step of the SA algorithm, as applied to the target domain. This PCA step results in a new coordinate system $\mathcal{X}_T$, defined by eigenvectors of the target data. These eigenvectors have an ambiguity, because if v_i is an eigenvector, then $-v_i$ is also an eigenvector. Which sign is used for a basis vector can depend on the implementation of PCA, and can affect the resulting classification accuracy.

We can remove this ambiguity if we have a small amount of labeled target domain data D_{TL}. To determine whether to flip the sign of each axis of $\mathcal{X}_T$, the algorithm can check the resulting accuracy on the labeled target data. Use this "sign flip" feature with the algorithm for the parts below.

The number of labeled target domain data is $N_{TL} = 20$.

(i) Use SA with the sign flip feature. Train and test the SA method on different θ.

(ii) For fair comparison, the training data of your SL counterpart should also contain D_{TL}. Train and test the SL classifier on different θ. (All in the original feature space, using standardized data.)

(iii) Draw plots of testing accuracy vs. θ for SA and SL on the same figure. Try the three Qs as in (b).

(iv) Which performs better, SA or SL? Does sign flipping and/or adding labeled target data help improve the performance of SL or SA? Why?

Note: For SA, just use D_{TL} for sign flipping, and the base classifier is still trained on source domain data only. For SL, add D_{TL} into the training set. What if we also add D_{TL} into the training set for Q in SA? Try it yourself and no need to report the results here.

(d) Use the algorithm of (c) and try N_{TL} in $\{0, 6, 20, 50, 100\}$ for Q=LDA.

(i) Draw the plot of SA testing accuracy vs. θ for different N_{TL} on the same plot.

(ii) Draw the plot of SL testing accuracy vs. θ for different N_{TL} on the same plot.

(iii) How does N_{TL} affect the final performance of SA and SL? Why?

(e) Suppose $\theta = 30°$, $N_{TL} = 20$, Q=LDA. Now we would like to see whether standardization makes a difference. For sign flipping, use the algorithm of (c); without sign flipping, use the algorithm of (b); in both cases, implement with and without standardization.

Report the testing accuracy of SA and SL, and put your testing accuracies in the table below. How does standardization affect the accuracy? Why?

Sign flipping	Standardization	SA	SL
√	×		
√	√		
×	×		
×	√		

2. This problem concerns the generalization error bound in a transfer learning problem, as given in Lecture 14, Eq. (6).

In this problem you will study the effects of varying N_S, N_T, and α on the cross-domain generalization error bound.

Throughout this problem, let $\varepsilon_{\alpha\beta}$ be everything in the cross-domain generalization-error bound (RHS of Lecture 14 Eq. (6)), except omitting $e^*_{S,T}$. Note that $e^*_{S,T}$ is a constant of the parameters we will be varying.

Also throughout this problem, use the values $d_{VC} = 10$, $\delta = 0.1$, $d_{\mathcal{H}\Delta\mathcal{H}} = 0.1$. However, leave them as variables until you are ready to plot, or until you are asked for a number.

(a) Give the simplified number (to two decimal digits) for $\varepsilon_{\alpha\beta}$, for the following cases:

(i) $N_T = 1,\ N_S = 100,\ \alpha = 0.1,\ 0.5,\ 0.9$

(ii) $N_T = 10,\ N_S = 1000,\ \alpha = 0.1,\ 0.5,\ 0.9$

(iii) $N_T = 100,\ N_S = 10000,\ \alpha = 0.1,\ 0.5,\ 0.9$

(iv) $N_T = 1000,\ N_S = 100000,\ \alpha = 0.1,\ 0.5,\ 0.9$

Tip: put these in a table for easy viewing.

(v) Do any of these sets of numbers assure some degree of generalization (*i.e.*, $\varepsilon_{\alpha\beta} < 0.5$, assuming $e^*_{S,T} \approx 0$)? If so, which?

Comment: As in the supervised learning case, these bounds can be very loose, but evidence indicates the functional dependence of $\varepsilon_{\alpha\beta}$ on its variables still generally apply.

(b) For this part, let $N_S = 1000$ and plot $\varepsilon_{\alpha\beta}$ vs. α for $N_T = 10,\ 100,\ 1000,\ 10000$ (4 curves on one plot), over $0 \leq \alpha \leq 1$. Answer: what approximate value of α is optimal for each value of N_T ? Try to explain the dependence of $\varepsilon_{\alpha\beta}$ on α for different values of N_T , and any difference in optimal values of α.

(c) For this part, let $N_T = 100$ and plot $\varepsilon_{\alpha\beta}$ vs. α for $N_S = 10,\ 100,\ 1000,\ 10000$ (4 curves on one plot), over $0 \leq \alpha \leq 1$. Answer: what approximate value of α is optimal for each value of N_T ? Try to explain the dependence of $\varepsilon_{\alpha\beta}$ on α for different values of N_S , and any difference in optimal values of α.

(d) Common default values for α are $\alpha = 0.5$ and $\alpha = \beta$.

(i) In terms of minimizing the cross-domain generalization-error bound, which default choice looks better (based on your answers to (b) and (c) above)? Is that choice reasonably consistent with your results of (b) and (c)?

(ii) Give algebraic expressions for $\varepsilon_{\alpha\beta}(\alpha = 0.5)$ and $\varepsilon_{\alpha\beta}(\alpha = \beta)$. Compare them algebraically: can you draw any conclusions about which is lower?

(iii) Plot $\varepsilon_{\alpha\beta}(\alpha = 0.5)$ vs. N for $\beta = 0.01, 0.1, 0.5$, for $1000 \leq N \leq 100000$ (3 curves on 1 plot). Repeat for $\varepsilon_{\alpha\beta}(\alpha = \beta)$. What conclusions can you draw from the plots?

(e) In this problem, do the generalization-error bounds you have calculated typically relate to training set error or test set error? Briefly justify your answer.