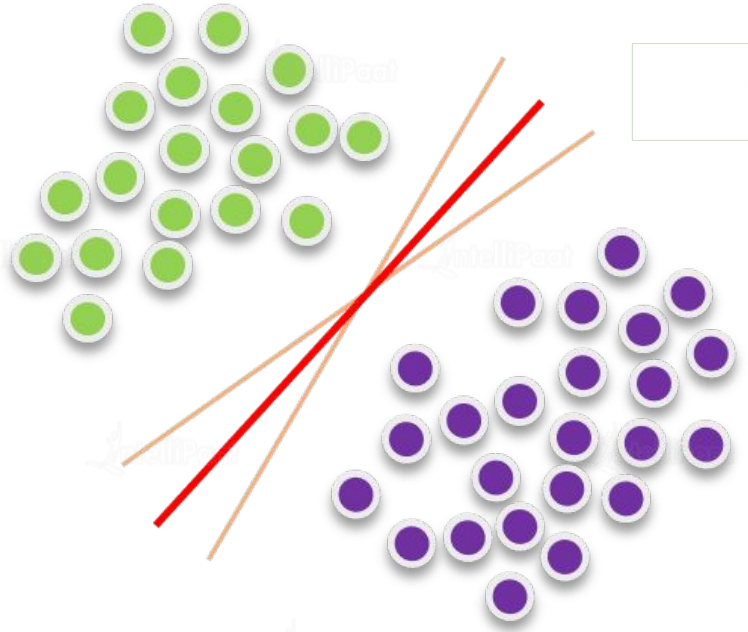
A Classification Method based on Generalized Eigenvalue Problems

*Implementation of **Regularized General Eigenvalue Classifier (ReGEC)** using R programming language*



Presented by:

1. Abdelrahman Usama Gabr Abdou Habib
2. Md Imran Hossain
3. Muhammad Zain Amin

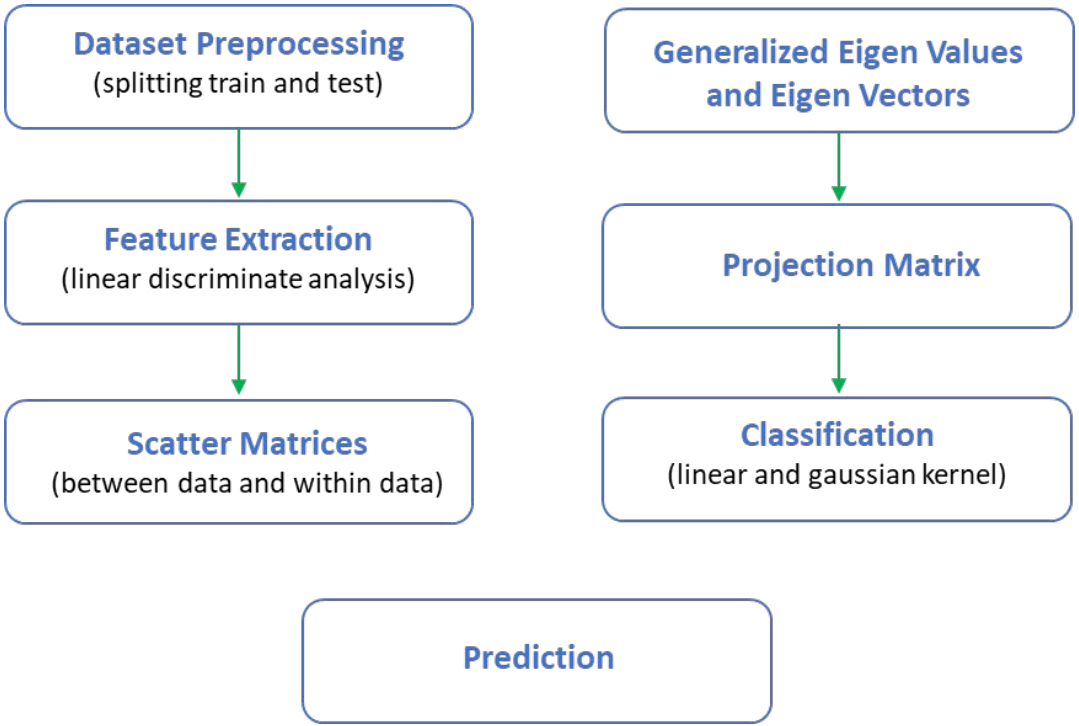
Introduction

- Regularized Generalized Eigenvalue Classifier (RGEC) is a supervised classification algorithm used in machine learning and pattern recognition. It is based on the idea of generalized eigenvalue problem and uses regularization to prevent overfitting.

- RGEC is a powerful classification algorithm that can be used for a wide range of applications in areas such as computer vision, speech recognition, and bioinformatics.

- R is a popular programming language for statistical computing and graphics, and it provides several packages that can be used to implement Regularized Generalized Eigenvalue Classifier (RGEC).

ReGEC Implementation Approach



R: Installed packages

- library(MASS)
- library(caret)
- library(e1071)

Data handling & Pre-processing

- For the experimentation and evaluation of our classifier, we have used four different datasets, train and test data splitted (90:10) for linear kernel and (70:30) for gaussian kernel.

1. Cleveland Heart
2. Pima Indians
3. Breast Cancer
4. German

- All of the above mentioned datasets have been imported from the UCI Machine Learning Repository and Data World.

Feature Extraction

- We have used the **Linear Discriminant Analysis (LDA)** for the extraction of features from the training and testing data.

Covariance or Scatter Matrix

- We have computed the overall mean and variance for train features in order to get Compute the between-class and within-class scatter matrices.

Generalized Eigenvalues and Eigenvectors

- We have computed the generalized eigenvectors and eigenvalues and sort the eigenvalues in descending order in order to get the maximum eigenvalue. The obtained eigenvectors help to calculate the projection matrix.

Classification: Gaussian and Linear Kernels

- We have used both the linear and gaussian kernel during the experimentation and evaluation of ReGEC classifier.

    - Linear Kernel was used for evaluating the ReGEC classifier on Cleveland and Pima datasets.
    - Gaussian Kernel was used for ReGEC classifier evaluation on Breast Cancer and German datasets.

Experimentation and Results (1 / 2)

- We found the results regard their performance in terms of classification accuracy and execution time.
  - The results regarding the linear kernel have been obtained using the Cleveland Heart and Pima Indians Database as shown in table 1.

Table 1. ReGEC classification accuracy using linear kernel.

Dataset	Train (Sample Size)	Test (Sample Size)	Number of Features	Accuracy (%)		Elapsed Time (sec)	
				Obtained	Paper	Obtained	Paper
Cleveland Heart	297	30	13	86.67	86.05	1.979e-03	1.92e-04
Pima Indians	768	77	8	80.52	74.91	7.021e-03	1.21e-04

Experimentation and Results (1 / 2)

- We found the results regard their performance in terms of classification accuracy and execution time.
  - The results regarding the linear kernel have been obtained using the Breast Cancer and German Database as shown in table 2.

Table 2. ReGEC classification accuracy using gaussian kernel.

Dataset	Train (Sample Size)	Test (Sample Size)	Number of Features	Lamda	Gamma	Accuracy (%)		Elapsed Time (sec)	
						Obtained	Paper	Obtained	Paper
Breast Cancer	200	77	9	1.e-03	50	77.91	73.40	4.783e-03	0.0698
German	700	300	8	1.e0-3	500	72.67	70.26	3.427e-02	3.8177

Conclusion

- ReGEC is a powerful classification algorithm that can be used for a wide range of applications. The regularization step is particularly important in preventing overfitting and improving the generalization performance of the classifier.

- The ReGEC classifier has shown better performance in terms of accuracy while using the linear kernel as compared to the gaussian kernel.

- Our approach using the combination of LDA and SVM has outperformed in classification tasks on 4 different datasets, in-terms of accuracy and elapsed time.