Adaptive LASSO

Introduction

This report presents a complete example of Adaptive Lasso Modelling, including full explanations of algebraic terms

Middle East and North Africa Data Set

The dataset comprises records of 39,063 events spanning various years, months, and days, each event documented across 13 distinct columns. These columns encapsulate critical attributes of the events, including the number of fatalities, the lethality classification, responsible groups, target types, attack methodologies, weapon implementations, and geographical information such as the country, province, and city of occurrence. With temporal, categorical, and geographical dimensions well-represented, the dataset is conducive to a wide array of analyses, including trend identification, geographical hotspot detection, risk assessment, policy formulation, and academic research across disciplines like criminology, sociology, and political science.

Middle East and North Africa Binary Data Set

The dataset consists of 39,063 rows and 38 columns. Each row represents a unique event, while the columns capture various attributes of these events. These attributes include temporal information such as the year, month, and day of each event, as well as categorical data such as the responsible group, target type, attack method, weapon used, and geographical details like the country, province, and city of occurrence. Each column is binary-encoded, indicating the presence or absence of specific categories within each attribute. This dataset is conducive to exploratory analyses aimed at understanding patterns and trends in events related to conflict or security incidents.

Data Cleaning

This section implements some data cleaning, specifically removing variables that have zero or near zero variance and highly correlated variables

Zero and near - zero variance features: these features refer to variables in a dataset that have either zero or very little variance. These features typically do not provide useful information for predictive modeling and may even hinder the performance of the model. Here's an explanation of both:

Zero Variance Features: these are variables that have the same value for all observations in the dataset. Because the variable does not vary at all, it cannot help differentiate between different observations. For example, if a variable has a value of 0 for all observations, it has zero variance.

Near-Zero Variance Features: these are variables that have very little variation in their values across the dataset. While they may have more than one unique value, one value might dominate the majority of the observations, making it nearly constant. Near-zero variance features are often undesirable because they don't provide enough information to the model and can lead to overfitting. Identifying and removing near-zero variance features can help simplify the model and improve its performance. In predictive modeling, it's essential to identify and handle zero and near-zero variance features appropriately. Removing these features can streamline the model, reduce complexity, and improve its ability to generalize to new data. Techniques such as univariate analysis, variance thresholds, and correlation analysis are commonly used to detect and address these issues.

There is no near zero or zero variance

Correlation

We use Spearman's rank correlation coefficient is used. Spearman correlation measures the strength and direction of association between two variables, and it's particularly useful for assessing monotonic relationships between variables. Here, any pair of variables with a correlation coefficient greater than or equal to 0.70 (in absolute value) will be considered highly correlated. The function returns a logical vector indicating which variables are highly correlated with each other. If two variables are highly correlated, one of them might be redundant for modeling purposes, and removing one can help avoid multicollinearity issues.

Predictions

Here we create training and test data. For Training Data, select all data upto and including 2019. We then choose the year 2020 to be predicted.

Lasso Modelling

Lasso modeling, or Lasso regression, is a type of linear regression that includes a regularization technique. Lasso stands for Least Absolute Shrinkage and Selection Operator. It is particularly useful when you have a large number of features (predictors) and want to both predict outcomes and perform feature selection.

Key Characteristics

Feature Selection: Lasso can shrink some coefficients to exactly zero, effectively performing feature selection by excluding non-informative features.

Sparse Models: The resulting models from Lasso regression are often sparse, meaning they use only a subset of the features. This makes the model easier to interpret.

Bias-Variance Tradeoff: By adding the regularization term, Lasso increases bias slightly but significantly reduces variance, leading to better overall performance, especially on unseen data. When to Use Lasso Regression

High-dimensional Data: When you have a large number of features and expect that only a small number of them are useful for predicting the output.

Feature Selection: When you want to automatically perform feature selection as part of the modeling process.

Preventing Overfitting: When you need to improve the generalization of your model by preventing it from fitting to the noise in the training data.

Key Concepts:

Linear Regression: A method for modeling the relationship between a dependent variable and one or more independent variables. It assumes that the dependent variable is a linear combination of the independent variables.

Regularization: A technique used to prevent overfitting by adding a penalty to the model's complexity. Regularization discourages the model from fitting the noise in the training data, thereby improving its generalization to new data.

L1 Regularization: The penalty used in Lasso regression. It adds the absolute values of the coefficients (weights) to the cost function.

Cost Function: The cost function in Lasso regression is modified to include the L1 penalty. The goal of the Adaptive Lasso is to minimize this cost function, balancing the goodness-of-fit (as measured by the RSS) and the complexity of the model (as controlled by the adaptive penalty). This approach enhances the ability of the model to generalize to new data by reducing overfitting and selecting the most important predictors.

$$\text{Cost Function} = \sum_{i=1}^{n}\left(y_i - \left(\beta_0 + \sum_{j=1}^{p}\beta_j\, x_{ij}\right)\right)^2 + \lambda\sum_{j=1}^{p}|\beta_j|$$

Explanation of Components

Residual Sum of Squares:

$$\sum_{i=1}^{n}\left(y_i - \left(\beta_0 + \sum_{j=1}^{p}\beta_j\, x_{ij}\right)\right)^2$$

This term is the **residual sum of squares (RSS)**, representing the fit of the model to the data. It measures the discrepancy between the observed values y_i and the values

predicted by the linear model $\beta_0 + \sum_{j=1}^{p} \beta_j x_{ij}$. The RSS component of the cost function is minimized when the predicted values are as close as possible to the actual response values. A smaller RSS indicates a better fit of the model to the data.

Adaptive Penalty:

$$\lambda \sum_{j=1}^{p} \frac{|\beta_j|}{|\hat{\beta}_j|^{\gamma}}$$

Here's what Adaptive Penalty means:

Regularization Parameter (λ): This is a tuning parameter that controls how much penalty we apply to the model's coefficients. A larger λ means more penalty, which makes the model simpler by shrinking more coefficients towards zero. A smaller λ means less penalty, allowing the model to fit the data more closely but potentially increasing the risk of overfitting.

Sum Over All Predictors ($\sum_{j=1}^{p}$): This symbol represents adding up the penalty for each predictor (or feature) in the model. The sum runs from $j = 1$ to $j = p$, where p is the total number of predictors.

Coefficient of Each Predictor (β_j): β_j is the coefficient for the j-th predictor in the model. The penalty is applied to the absolute value of these coefficients.

Initial Estimates of Coefficients ($\hat{\beta}_j$): $\hat{\beta}_j$ is the initial estimate of the coefficient for the j-th predictor, usually obtained from a preliminary model (like Ordinary Least Squares). These initial estimates help determine how much penalty to apply to each coefficient in the final model.

Exponent (γ): γ is a positive constant, typically set to 1. It adjusts the influence of the initial estimates on the penalty. When γ is 1, the penalty for each coefficient is directly proportional to the inverse of the initial estimate's absolute value.

This term is the **adaptive penalty**, where λ is a regularization parameter that controls the amount of shrinkage applied to the coefficients. The penalty term helps prevent overfitting by shrinking some coefficients towards zero, thereby performing feature selection. By incorporating these adaptive weights, the Adaptive Lasso can more accurately select relevant features and improve prediction accuracy compared to standard Lasso regression.

Regular Lasso Modelling

```
cv_model_Year_Lasso <- cv.glmnet(x_Year_Lasso,
                                 y_Year,
                                 alpha = 1,
                                 nfolds = 10)
```

This code above performs k-fold cross-validation for the Lasso regression model using the cv.glmnet function from the glmnet package. Here's a breakdown of each component:

cv_model_Year_Lasso: This variable stores the result of the cross-validated Lasso regression model.

cv.glmnet(): This function conducts cross-validation for a Lasso (or elastic net) regression model. It takes several arguments:

x_Year_Lasso: The predictor variables (in matrix form) for the model.

y_Year: The response variable (dependent variable) for the model.

alpha = 1: Specifies that Lasso regularization is used. An alpha value of 1 corresponds to the Lasso penalty, while an alpha value of 0 would correspond to the ridge penalty.

nfolds = 10: Specifies the number of folds for cross-validation. In this case, it performs 10-fold cross-validation, meaning the data is divided into 10 subsets, and the model is trained and tested 10 times, each time using a different subset as the test set.

Cross-validation is a technique used to assess how well a predictive model will generalize to an independent dataset. In k-fold cross-validation, the dataset is randomly partitioned into k equal-sized subsamples. Of the k subsamples, a single subsample is retained as the validation data for testing the model, and the remaining k-1 subsamples are used as training data. The cross-validation process is then repeated k times (the folds), with each of the k subsamples used exactly once as the validation data. The k results are then averaged to produce a single estimation. The advantage of this method is that all observations are used for both training and validation, and each observation is used for validation exactly once.

Optimal lambda value that minimizes test MSE

```
best_lambda_Year_Lasso <- cv_model_Year_Lasso$lambda.min
```

This code above retrieves the optimal value of the regularization parameter lambda selected by the cross-validated Lasso regression model.

cv_model_Year_Lasso$lambda.min: This part accesses the lambda.min component of the cv_model_Year_Lasso object.

The cv_model_Year_Lasso object stores the results of cross-validation performed on the Lasso regression model.

The lambda.min component represents the value of lambda that minimizes the cross-validated error. This lambda value is determined during the cross-validation process as the optimal regularization parameter for the model.

best_lambda_Year_Lasso: This variable stores the value of the optimal lambda parameter, which is selected based on the cross-validation results. This lambda value is then used to fit the final Lasso regression model.

Improved Lasso Model

This code fits a Lasso regression model to the training data using the optimal value of the regularization parameter lambda determined through cross-validation.

glmnet: This function is used to fit a generalized linear model via penalized maximum likelihood. It is specifically designed for fitting regularized regression models, such as Lasso and Ridge regression.

x_Year_Lasso: This is the matrix of predictor variables used to train the model. y_Year: This is the response variable used to train the model. alpha = 1: The alpha parameter specifies the type of penalty term used in the model. In this case, alpha = 1 indicates that the Lasso penalty (L1 regularization) is applied.

nfolds = 10: This parameter specifies the number of folds used in cross-validation. The data is divided into 10 folds, and the model is trained and tested iteratively on different combinations of folds. lambda = best_lambda_Year_Lasso: This parameter specifies the value of the regularization parameter lambda to be used in the model. best_lambda_Year_Lasso is the optimal value of lambda selected through cross-validation, which minimizes the cross-validated error.

A better version of LASSO: This comment indicates that the Lasso regression model with the optimal lambda value selected through cross-validation is expected to perform better in terms of predictive accuracy compared to other versions of the Lasso model with different lambda values.

```
best_model_Year_Lasso <- glmnet(x_Year_Lasso,
                                y_Year,
                                alpha = 1,
                                nfolds = 10,
                                lambda = best_lambda_Year_Lasso) # a better v
ersion of LASSO
```

Relaxed Lasso Modelling

Here's an explanation of the code:

glmnet: As before, this function is used to fit a generalized linear model via penalized maximum likelihood, specifically for regularized regression models like Lasso.

x_Year_Lasso: Matrix of predictor variables used to train the model.

y_Year: Response variable (Lethal attack) used to train the model.

alpha = 1: Specifies that the Lasso penalty (L1 regularization) is applied.

nfolds = 10: Specifies the number of folds used in cross-validation.

lambda = best_lambda_Year_Lasso: Specifies the value of the regularization parameter lambda to be used in the model. best_lambda_Year_Lasso is the optimal lambda value selected through cross-validation, which minimizes the cross-validated error.

relax = TRUE: This additional argument tells the glmnet function to relax the Lasso penalty for certain variables. Relaxing the penalty means that these variables are treated less strictly in terms of regularization, allowing them to have larger coefficients in the model. This can be useful when dealing with multicollinearity or when there are groups of highly correlated predictors. By relaxing the penalty for some variables, the model may achieve better predictive performance.

```
best_model_Year_Lasso_relaxed <- glmnet(x_Year_Lasso,
                                        y_Year,
                                        alpha = 1,
                                        nfolds = 10,
                                        lambda = best_lambda_Year_Lasso,
                                        relax = TRUE) # a better version of L
ASSO
```

Perform Adaptive Lasso

```
best_model_Year_Lasso_adaptive <- cv.glmnet(
                                  x = x_Year_Lasso,
                                  y = y_Year,
                                  alpha = 1,
                                  penalty.factor = 1 / abs(best_ridge_coef))
```

This code snippet above performs adaptive LASSO regularization using cross-validation to find the optimal penalty parameter (lambda). Here's an explanation of the parameters:

x: The matrix of predictor variables.

y: The response variable (Lethal Attack).

alpha: The elastic net mixing parameter. When alpha = 1, it represents the LASSO penalty, which encourages sparsity in the coefficients.

penalty.factor: A vector of length equal to the number of predictors, where each element represents a separate penalty factor that can be applied to each coefficient.

In adaptive LASSO, the penalty factors are adjusted based on the ridge regression coefficients (best_ridge_coef), with the goal of providing differential shrinkage to different variables. This means that some variables may be more heavily penalized than others, allowing the model to perform variable selection and regularization more effectively. Overall, this code is setting up a cross-validated adaptive LASSO regression model, where

the regularization strength is adaptively adjusted for each predictor based on its estimated coefficients from ridge regression.

```
## 25 x 1 sparse Matrix of class "dgCMatrix"
##                                s1
## (Intercept)          0.775915275
## OtherGroup          -0.094587373
## Business             .
## GovtGen             -0.123719374
## OtherTarget         -0.098153449
## Police               .
## Private              .
## ArmedAssaultAttack   .
## Assassination        0.057627309
## HostageKidnapAttack -0.345683143
## OtherAttack         -0.223026862
## Firearms             0.271709097
## OtherWeapon          0.120468858
## OtherCountry         .
## Syria                0.195129994
## Israel_Nationality  -0.272014416
## OtherNationality    -0.124405931
## Turkey_Nationality  -0.169896524
## Yemen_Nationality   -0.092428583
## Al_Anbar_Province    .
## Diyala_Province      0.002960996
## Nineveh_Province    -0.011450674
## OtherProvince       -0.035401487
## Saladin_Province     .
## OtherCity           -0.085934712
```

The output represents the coefficients from a Lasso regression model, highlighting which predictor variables have an impact on the dependent variable and the direction of that impact.

Key Points:

Intercept (0.775915275): Baseline value of the dependent variable.

Non-zero Coefficients: Indicate significant predictors:

Negative Associations: OtherGroup, GovtGen, OtherTarget, HostageKidnapAttack, OtherAttack, Israel_Nationality, OtherNationality, Turkey_Nationality, Yemen_Nationality, Nineveh_Province, OtherProvince, OtherCity.

Positive Associations: Assassination, Firearms, OtherWeapon, Syria, Diyala_Province.

Zero Coefficients: Variables with no significant impact, effectively excluded by the model: Business, Police, Private, Al_Anbar_Province, Saladin_Province.

Summary:

Lasso regression identified key predictors by setting some coefficients to zero, indicating no significant impact, while retaining others with non-zero values, indicating their importance and direction of influence on the dependent variable.