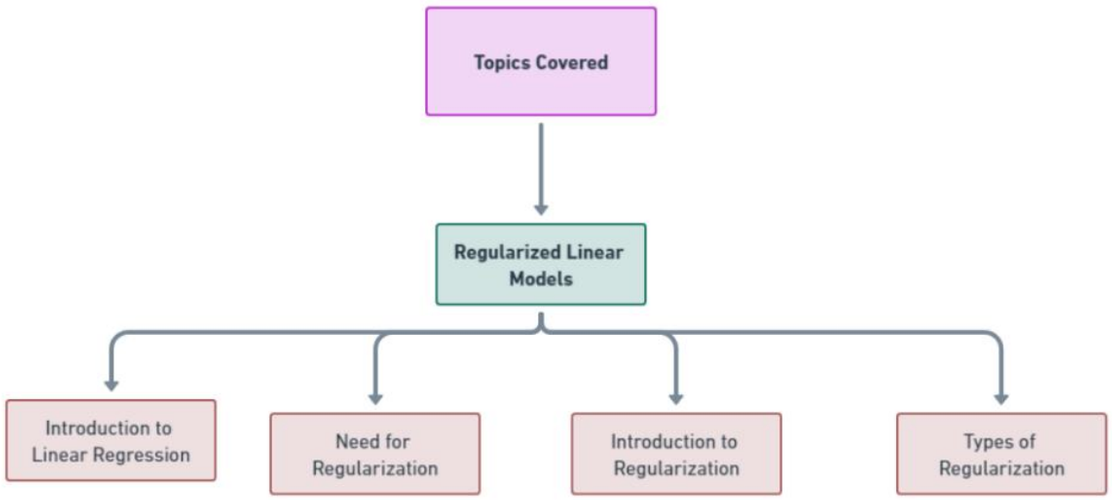
PW SKILLS

Lesson Plan

Regularized Linear MoRidge & Lasso Regressiondels

Topic's Covered

- Introduction to Regression
- Need for Regularization
- L2 Regularization
  - Advantages of Ridge Regression
  - Limitations of Ridge Regression
  - Practical Implementation (Ridge)
- L1 Regularization
  - Advantages of Lasso Regression
  - Limitations of Lasso Regression
  - Practical Implementation (Lasso)



- The primary goal is to understand and quantify the patterns within data, allowing us to make predictions or infer insights.
- Regression models are widely employed in various fields, including finance, economics, biology, and engineering.

Types of Regression

- **Linear Regression:**
  - Linear regression assumes a linear relationship between the input variables and the target variable.
  - It aims to fit a straight line to the data points, minimizing the sum of squared differences between the observed and predicted values.
- **Polynomial Regression:**
  - Polynomial regression extends the linear model by introducing polynomial features.
  - It can capture non-linear relationships by incorporating terms with higher degrees, providing a more flexible fit to the data.
- **Ridge Regression:**
  - Ridge regression is a type of linear regression that includes a regularization term to prevent overfitting.
  - It is particularly useful when dealing with multicollinearity, where independent variables are highly correlated.
- **Lasso Regression:**
  - Lasso regression, similar to Ridge, introduces a regularization term. However, Lasso employs L1 regularization, which tends to produce sparse models by driving some of the coefficient values to exactly zero.

Need for Regularization

- **Overfitting and Underfitting:**
  - Overfitting: Occurs when a model learns the training data too well, capturing noise and fluctuations rather than the underlying pattern. This leads to poor generalization performance on new, unseen data.
  - Underfitting: Happens when a model is too simple to capture the underlying patterns in the data. It fails to learn the training data adequately and performs poorly on both the training and test datasets.
- **Bias-Variance Tradeoff:**
  - The bias-variance tradeoff is a crucial concept in machine learning that illustrates the balance between bias and variance in the model.
  - Bias: Refers to the error introduced by approximating a real-world problem with a simplified model. High-bias models are too simplistic and may underfit the data.
  - Variance: Represents the model's sensitivity to small fluctuations in the training data. High-variance models are complex and can capture noise, leading to overfitting.

L2 Regularization (or) Ridge regression

- In machine learning, Ridge Regression, also known as L2 regularization, is a technique employed to address the issue of overfitting in linear regression models.
- Unlike traditional linear regression, Ridge Regression adds a regularization term to the cost function, aiming to penalize the model for excessively large coefficients.
- We can quantify complexity using the L2 regularization formula, which defines the regularization term as the sum of the squares of all the feature weights:

$$L_2 \text{ regularization term} = ||\boldsymbol{w}||_2^2 = w_1^2 + w_2^2 + \ldots + w_n^2$$

- In this formula, weights close to zero have little effect on model complexity, while outlier weights can have a huge impact.
- For example, a linear model with the following weights:

$$\{w_1 = 0.2, w_2 = 0.5, w_3 = 5, w_4 = 1, w_5 = 0.25, w_6 = 0.75\}$$

$$w_1^2 + w_2^2 + \boldsymbol{w_3^2} + w_4^2 + w_5^2 + w_6^2$$

$$= 0.2^2 + 0.5^2 + \boldsymbol{5^2} + 1^2 + 0.25^2 + 0.75^2$$

$$= 0.04 + 0.25 + \boldsymbol{25} + 1 + 0.0625 + 0.5625$$

$$= 26.915$$

- **Handling Multicollinearity:**
  - Multicollinearity occurs when two or more predictor variables in a regression model are highly correlated, leading to instability and inflated standard errors in coefficient estimates. Ridge Regression is particularly effective in addressing multicollinearity.
  - By introducing the regularization term, Ridge Regression modifies the optimization objective to minimize both the mean squared error and the sum of squared coefficients. This has the practical effect of shrinking the coefficients of correlated variables towards each other, effectively reducing their impact on the model. As a result, Ridge Regression provides a more stable solution in the presence of multicollinearity.

Advantages of Ridge Regression

- Ridge Regression is effective in handling multicollinearity, a situation where predictor variables are highly correlated. It adds a penalty term to the coefficients, preventing them from becoming too large.
- Ridge Regression provides more stable estimates of the coefficients compared to ordinary least squares (OLS) regression, especially when dealing with a dataset with high dimensionality or a small sample size.
- It helps in preventing overfitting by adding a regularization term to the cost function, which penalizes large coefficients. This is particularly useful when dealing with complex models.
- Ridge Regression performs well when the input data matrix is ill-conditioned or when some variables are highly correlated. It adds regularization, making the problem more well-posed.
- Ridge Regression is less sensitive to outliers compared to plain linear regression. The regularization term helps mitigate the impact of extreme values.

Limitations of Ridge Regression

- Ridge Regression does not perform variable selection. It will include all the features in the model, although it may shrink their coefficients close to zero. If true feature selection is required, other methods like Lasso Regression might be more suitable.
- The introduction of the regularization term can make interpretation of coefficients less straightforward compared to traditional linear regression. It might be challenging to explain the impact of a specific variable on the response variable.
- If the true underlying model is sparse (i.e., only a small subset of features truly contribute), Ridge Regression might not perform as well as Lasso Regression, which has a tendency to drive some coefficients exactly to zero.

Practical Implementation

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import Ridge
from sklearn.metrics import mean_squared_error
from sklearn.datasets import load_diabetes

# Load the Diabetes dataset
diabetes = load_diabetes()
data = pd.DataFrame(data=diabetes.data,
columns=diabetes.feature_names)
target = diabetes.target

# Split the dataset into training and testing sets

X_train, X_test, y_train, y_test = train_test_split(data, target,
test_size=0.2, random_state=42)

# Initialize the Ridge Regression model
ridge_model = Ridge(alpha=1.0)  # Alpha is the regularization
parameter

# Fit the model to the training data
ridge_model.fit(X_train, y_train)
```

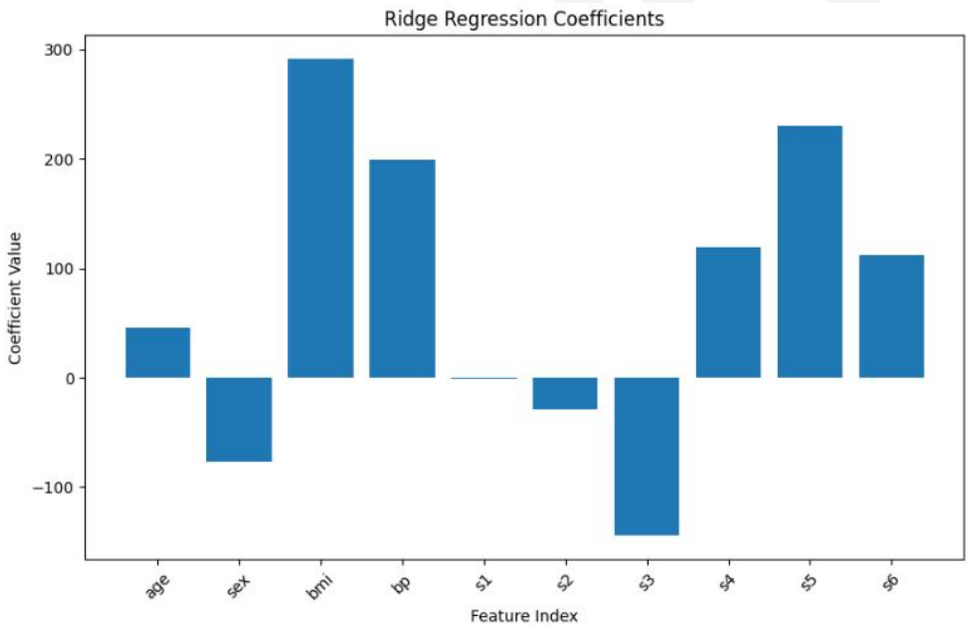
```
# Make predictions on the test set
y_pred = ridge_model.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
print(f'Mean Squared Error: {mse}')

# Display the coefficients
coefficients = ridge_model.coef_

# Plot the coefficients
plt.figure(figsize=(10, 6))
plt.bar(range(len(coefficients)), coefficients)
plt.xlabel('Feature Index')
plt.ylabel('Coefficient Value')
plt.title('Ridge Regression Coefficients')
plt.xticks(range(len(coefficients)), data.columns, rotation=45)
plt.show()
```

Output:



Lasso Regression (or) L1 Regularization

- Lasso Regression, short for "Least Absolute Shrinkage and Selection Operator," is a type of linear regression technique that incorporates regularization to prevent overfitting and perform feature selection.
- In Lasso Regression, the standard linear regression objective is modified by adding a penalty term, specifically the L1 norm of the coefficient vector.
- In simpler terms, Lasso Regression adds a constraint to the linear regression problem, limiting the size of the coefficients. This constraint leads to some coefficients becoming exactly zero, effectively performing automatic feature selection.
- The objective function of Lasso Regression is a combination of two components: the mean squared error (MSE) term and the L1 regularization term. The objective function aims to minimize the errors in prediction while simultaneously penalizing the sum of the absolute values of the coefficients.
- The L1 regularization term is added to the ordinary least squares (OLS) cost function, and the resulting cost function is defined as:

$$J(\theta) = \text{OLS Cost Function} + \lambda \sum_{i=1}^{n} |\theta_i|$$

Advantages of Lasso Regression

- Lasso Regression encourages sparsity, leading to automatic feature selection.
- Lasso helps prevent overfitting by penalizing the absolute values of the coefficients. This regularization term aids in controlling the complexity of the model, making it more generalizable to new, unseen data.
- Lasso Regression is effective in handling multicollinearity (high correlation between predictor variables). It can arbitrarily select one variable from a group of correlated variables and assign nonzero coefficients, simplifying the model.
- Lasso can be a useful tool in feature engineering, helping to identify and retain the most relevant features. This is especially valuable when working with large datasets with numerous potential predictor variables.

Limitations of Lasso Regression

- Lasso may exhibit instability when faced with highly correlated features or when there are groups of variables that are correlated. In such cases, it might arbitrarily select one variable over another, leading to variability in variable selection.
- The solution path of Lasso Regression is dependent on the order in which variables enter the model. This means that if two variables have similar levels of importance, the one entering the model first may be selected while the other is excluded.
- Lasso Regression can be sensitive to outliers in the data. Outliers may disproportionately influence the optimization process, leading to biased coefficient estimates.

Practical Implementation

```
# Import necessary libraries
import numpy as np
import pandas as pd
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.linear_model import Lasso
from sklearn.metrics import mean_squared_error
import matplotlib.pyplot as plt

# Load the diabetes dataset
diabetes = datasets.load_diabetes()
X = diabetes.data
y = diabetes.target

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

# Initialize the Lasso Regression model
lasso_model = Lasso(alpha=0.01)  # You can adjust the alpha
parameter for regularization strength

# Fit the model to the training data
lasso_model.fit(X_train, y_train)
```

```
# Make predictions on the test set
y_pred = lasso_model.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
print(f'Mean Squared Error: {mse}')

# Display the coefficients
coefficients = pd.DataFrame({'Feature': diabetes.feature_names,
'Coefficient': lasso_model.coef_})
print(coefficients)

# Plot the coefficients
plt.figure(figsize=(10, 6))
plt.bar(coefficients['Feature'], coefficients['Coefficient'])
plt.xlabel('Features')
plt.ylabel('Coefficient Value')
plt.title('Lasso Regression Coefficients')
plt.show()
```

Output: