Types of Regression Techniques in ML

A regression problem is when the output variable is a real or continuous value, such as “salary” or “weight”. Many different models can be used, the simplest is linear regression. It tries to fit data with the best hyperplane that goes through the points.

What is Regression Analysis?

Regression Analysis is a statistical process for estimating the relationships between the dependent variables or criterion variables and one or more independent variables or predictors. Regression analysis is generally used when we deal with a dataset that has the target variable in the form of continuous data. Regression analysis explains the changes in criteria about changes in select predictors. The conditional expectation of the criteria is based on predictors where the average value of the dependent variables is given when the independent variables are changed. Three major uses for regression analysis are determining the strength of predictors, forecasting an effect, and trend forecasting.

What is the purpose of using Regression Analysis?

There are times when we would like to analyze the effect of different independent features on the target or what we say dependent features. This helps us make decisions that can affect the target variable in the desired direction. Regression analysis is heavily based on statistics and hence gives quite reliable results to this reason only regression models are used to find the linear as well as non-linear relation between the independent and the dependent or target variables.

Types of Regression Techniques

Along with the development of the machine learning domain regression analysis techniques have gained popularity as well as developed manifold from just y = mx + c. There are several types of regression techniques, each suited for different types of data and different types of relationships. The main types of regression techniques are:

1. Linear Regression
2. Polynomial Regression
3. Stepwise Regression
4. Decision Tree Regression
5. Random Forest Regression
6. Support Vector Regression
7. Ridge Regression

8. Lasso Regression
9. ElasticNet Regression
10. Bayesian Linear Regression

Linear Regression

Linear regression is used for predictive analysis. Linear regression is a linear approach for modeling the relationship between the criterion or the scalar response and the multiple predictors or explanatory variables. Linear regression focuses on the conditional probability distribution of the response given the values of the predictors. For linear regression, there is a danger of overfitting. The formula for linear regression is:

> Syntax:
>
> $y = \theta x + b$
>
> where,
>
> - θ – It is the model weights or parameters
> - b – It is known as the bias.

This is the most basic form of regression analysis and is used to model a linear relationship between a single dependent variable and one or more independent variables.

Here, a linear regression model is instantiated to fit a linear relationship between input features (X) and target values (y). This code is used for simple demonstration of the approach.

Python

```
from sklearn.linear_model import LinearRegression

model = LinearRegression()

model.fit(X, y)

y_pred = model.predict(X_new)
```

Note: This code demonstrates the basic workflow of creating, training, and utilizing a linear regression model for predictive modeling tasks.

Polynomial Regression

This is an extension of linear regression and is used to model a non-linear relationship between the dependent variable and independent variables. Here as well syntax remains the same but now in the

input variables we include some polynomial or higher degree terms of some already existing features as well. Linear regression was only able to fit a linear model to the data at hand but with polynomial features, we can easily fit some non-linear relationship between the target as well as input features.

Here is the code for simple demonstration of the Polynomial regression approach.

Python

```
from  sklearn.linear_model  import  PolynomialRegression

model  =  PolynomialRegression(degree``=``2``)

model.fit(X, y)

y_pred  =  model.predict(X_new)
```

Note: This code demonstrates the basic workflow of creating, training, and utilizing a Polynomial regression model for predictive modeling tasks.

Stepwise Regression

Stepwise regression is used for fitting regression models with predictive models. It is carried out automatically. With each step, the variable is added or subtracted from the set of explanatory variables. The approaches for stepwise regression are forward selection, backward elimination, and bidirectional elimination. The formula for stepwise regression is

$$b_{j.std} = b_j(s_x s_y^{-1})$$

Here is the code for simple demonstration of the stepwise regression approach.

Python

```
from  sklearn.linear_model  import  StepwiseLinearRegression

model  =  StepwiseLinearRegression(forward``=``True``,

                    backward``=``True``,

                    verbose``=``1``)

model.fit(X, y)

y_pred  =  model.predict(X_new)
```

Note: This code demonstrates the basic workflow of creating, training, and utilizing a Stepwise regression model for predictive modeling tasks.

Decision Tree Regression

A Decision Tree is the most powerful and popular tool for classification and prediction. A Decision tree is a flowchart-like tree structure, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (terminal node) holds a class label. There is a non-parametric method used to model a decision tree to predict a continuous outcome.

Here is the code for simple demonstration of the Decision Tree regression approach.

Python

```
from  sklearn.tree  import  DecisionTreeRegressor

model  =  DecisionTreeRegressor()

model.fit(X, y)

y_pred  =  model.predict(X_new)
```

Note: This code demonstrates the basic workflow of creating, training, and utilizing a Decision Tree regression model for predictive modeling tasks.

Random Forest Regression

Random Forest is an ensemble technique capable of performing both regression and classification tasks with the use of multiple decision trees and a technique called Bootstrap and Aggregation, commonly known as bagging. The basic idea behind this is to combine multiple decision trees in determining the final output rather than relying on individual decision trees.

Random Forest has multiple decision trees as base learning models. We randomly perform row sampling and feature sampling from the dataset forming sample datasets for every model. This part is called Bootstrap.

Here is the code for simple demonstration of the Random Forest regression approach.

Python

```
from  sklearn.ensemble  import  RandomForestRegressor

model  =  RandomForestRegressor(n_estimators``=``100``)
```

```
model.fit(X, y)

y_pred = model.predict(X_new)
```

Note: This code demonstrates the basic workflow of creating, training, and utilizing a Random Forest regression model for predictive modeling tasks.

Support Vector Regression (SVR)

Support vector regression (SVR) is a type of support vector machine (SVM) that is used for regression tasks. It tries to find a function that best predicts the continuous output value for a given input value.

SVR can use both linear and non-linear kernels. A linear kernel is a simple dot product between two input vectors, while a non-linear kernel is a more complex function that can capture more intricate patterns in the data. The choice of kernel depends on the data's characteristics and the task's complexity.

Here is the code for simple demonstration of the Support vector regression approach.

Python

```
from sklearn.svm import SVR

model = SVR(kernel``=``'linear'``)

model.fit(X, y)

y_pred = model.predict(X_new)
```

Note: This code demonstrates the basic workflow of creating, training, and utilizing a Support vector regression model for predictive modeling tasks.

Ridge Regression

Ridge regression is a technique for analyzing multiple regression data. When multicollinearity occurs, least squares estimates are unbiased. This is a regularized linear regression model, it tries to reduce the model complexity by adding a penalty term to the cost function. A degree of bias is added to the regression estimates, and as a result, ridge regression reduces the standard errors.

$$\text{Cost} = \underset{\beta \in \mathbb{R}}{\operatorname{argmin}} \|i - X\beta\|^2 + \lambda \|\beta\|^2$$

Here is the code for simple demonstration of the Ridge regression approach.

Python

```
from  sklearn.linear_model  import  Ridge

model  =  Ridge(alpha``=``0.1``)

model.fit(X, y)

y_pred  =  model.predict(X_new)
```

Note: This code demonstrates the basic workflow of creating, training, and utilizing a Ridge regression model for predictive modeling tasks.

Lasso Regression

Lasso regression is a regression analysis method that performs both variable selection and regularization. Lasso regression uses soft thresholding. Lasso regression selects only a subset of the provided covariates for use in the final model.

This is another regularized linear regression model, it works by adding a penalty term to the cost function, but it tends to zero out some features' coefficients, which makes it useful for feature selection.

Here is the code for simple demonstration of the Lasso regression approach.

Python

```
from  sklearn.linear_model  import  Lasso

model  =  Lasso(alpha``=``0.1``)

model.fit(X, y)

y_pred  =  model.predict(X_new)
```

Note: This code demonstrates the basic workflow of creating, training, and utilizing a Lasso regression model for predictive modeling tasks.

ElasticNet Regression

Linear Regression suffers from overfitting and can't deal with collinear data. When there are many features in the dataset and even some of them are not relevant to the predictive model. This makes the model more complex with a too-inaccurate prediction on the test set (or overfitting). Such a model with high variance does not generalize on the new data. So, to deal with these issues, we

include both L-2 and L-1 norm regularization to get the benefits of both Ridge and Lasso at the same time. The resultant model has better predictive power than Lasso. It performs feature selection and also makes the hypothesis simpler. The modified cost function for Elastic-Net Regression is given below:

$$\frac{1}{m}\left[\sum_{l=1}^{m}\left(y^{(i)}-h\left(x^{(i)}\right)\right)^{2}+\lambda_1\sum_{j=1}^{n}w_j+\lambda_2\sum_{j=1}^{n}w_j^2\right]$$

where,

- **w(j)** represents the weight for the jth feature.
- **n** is the number of features in the dataset.
- **lambda1** is the regularization strength for the L1 norm.
- **lambda2** is the regularization strength for the L2 norm.

Here is the code for simple demonstration of the Elasticnet regression approach.

Python

```
from  sklearn.linear_model  import  ElasticNet

model  =  ElasticNet(alpha``=``0.1``, l1_ratio``=``0.5``)

model.fit(X, y)

y_pred  =  model.predict(X_new)
```

Note: This code demonstrates the basic workflow of creating, training, and utilizing a Elastic Net regression model for predictive modeling tasks.

Bayesian Linear Regression

As the name suggests this algorithm is purely based on Bayes Theorem. Because of this reason only we do not use the Least Square method to determine the coefficients of the regression model. So, the technique which is used here to find the model weights and parameters relies on features posterior distribution and this provides an extra stability factor to the regression model which is based on this technique.

Here is the code for simple demonstration of the Bayesian Linear regression approach.

Python

```
from sklearn.linear_model import BayesianLinearRegression

model = BayesianLinearRegression()

model.fit(X, y)

y_pred = model.predict(X_new)
```

Note: This code demonstrates the basic workflow of creating, training, and utilizing a Bayesian linear regression model for predictive modeling tasks.

Frequently Asked Questions(FAQs)

1.What are the 2 main types of regression?

> The two main types of regression are linear regression and logistic regression. Linear regression is used to predict a continuous numerical outcome, while logistic regression is used to predict a binary categorical outcome (e.g., yes or no, pass or fail).

2. What are the two types of variables in regression?

> The two types of variables in regression are independent variables and dependent variables. Independent variables are the inputs to the regression model, while the dependent variable is the output that the model is trying to predict.

3.Why is regression called regression?

> The term “regression” was coined by Sir Francis Galton in the late 19th century. He used the term to describe the phenomenon of children’s heights tending to regress towards the mean of the population, meaning that taller-than-average parents tend to have children who are closer to the average height, and shorter-than-average parents tend to have children who are closer to the average height.

4. How to calculate regression?

> There are many different ways to calculate regression, but the most common method is gradient descent. Gradient descent is an iterative algorithm that updates the parameters of the regression model in the direction that minimizes the error between the predicted and actual values of the dependent variable.

5. Why use regression?

> Regression is a powerful tool for understanding and predicting relationships between variables. It is used in a wide variety of applications, including finance, economics, marketing, and

> medicine.