17th April Assignment

April 29, 2023

1 Assignment 72

Q1. What is Gradient Boosting Regression?

Ans.Gradient Boosting Regression is a machine learning algorithm that is used for regression tasks. It is an ensemble method that combines multiple weak predictive models to create a strong predictive model. The algorithm works by iteratively adding new models to the ensemble, with each new model focusing on the errors of the previous models.

The term "gradient boosting" refers to the way the algorithm minimizes the loss function. At each iteration, the algorithm calculates the negative gradient of the loss function with respect to the output of the previous model. This gradient represents the direction in which the loss function is steepest, and the algorithm then fits a new model to the gradient

Q2. Implement a simple gradient boosting algorithm from scratch using Python and NumPy. Use a simple regression problem as an example and train the model on a small dataset. Evaluate the model's performance using metrics such as mean squared error and R-squared.

[1]:
```
# import dependencies
from sklearn.datasets import make_regression
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.model_selection import train_test_split
from sklearn.ensemble import GradientBoostingRegressor
```

[2]:
```
# Create dataset
X,y = make_regression(n_samples=1000,n_features=10,random_state=0,shuffle=False)
```

[3]:
```
# train. test and split
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.20)
```

[4]:
```
# train model
regressor = GradientBoostingRegressor()
regressor.fit(X_train,y_train)
```

[4]:
```
GradientBoostingRegressor()
```

[5]:
```
y_pred = regressor.predict(X_test)
```

[6]:
```
# Evaluate model
print('MSE: ',mean_squared_error(y_test,y_pred))
print('R2 score: ',r2_score(y_test,y_pred))
```

```
MSE:  1754.782236252065
R2 score:  0.9000004956356449
```

Q3. Experiment with different hyperparameters such as learning rate, number of trees, and tree depth to optimise the performance of the model. Use grid search or random search to find the best hyperparameters

[7]:
```
# import dependencies
from sklearn.datasets import make_regression
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.tree import DecisionTreeRegressor
```

[8]:
```
X,y = make_regression(n_samples=1000,n_features=10,random_state=0,shuffle=False)
```

[9]:
```
# train. test and split
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.20)
```

[10]:
```
# Define parameter grid for grid search
param_grid = {
    'n_estimators': [50, 100, 200],
    'learning_rate': [0.05, 0.1, 0.2],
    'max_depth': [2, 3, 4]
}
```

[11]:
```
regressor = GradientBoostingRegressor()
grid =␣
 ↪GridSearchCV(regressor,param_grid=param_grid,scoring='neg_mean_squared_error')
```

[12]:
```
# training data
grid.fit(X_train,y_train)
```

[12]:
```
GridSearchCV(estimator=GradientBoostingRegressor(),
             param_grid={'learning_rate': [0.05, 0.1, 0.2],
                         'max_depth': [2, 3, 4],
                         'n_estimators': [50, 100, 200]},
             scoring='neg_mean_squared_error')
```

[13]:
```
# best params
grid.best_params_
```

```
[13]: {'learning_rate': 0.2, 'max_depth': 2, 'n_estimators': 200}
```

```
[14]: # train model with hyperparameter tunning
      regressor = GradientBoostingRegressor(**grid.best_params_)
```

```
[15]: regressor.fit(X_train,y_train)
```

```
[15]: GradientBoostingRegressor(learning_rate=0.2, max_depth=2, n_estimators=200)
```

```
[16]: y_pred = regressor.predict(X_test)
```

```
[17]: # Evaluate model
      print('MSE: ',mean_squared_error(y_test,y_pred))
      print('R2 score: ',r2_score(y_test,y_pred))
```

```
MSE:  1194.7563364005882
R2 score:  0.9423124067098769
```

Q4. What is a weak learner in Gradient Boosting?

Ans.In Gradient Boosting, a weak learner is a simple model that is not very accurate on its own but can still contribute to the overall predictive power of the model. In the context of Gradient Boosting Regression, weak learners are typically decision trees with a small number of splits, also known as decision stumps.

Q5. What is the intuition behind the Gradient Boosting algorithm?

Ans.The intuition behind the Gradient Boosting algorithm is to combine multiple weak models in order to create a strong model that is able to capture complex relationships between the input features and the target variable.

The algorithm works by iteratively adding weak learners to the model, with each weak learner trying to correct the errors made by the previous ones. At each iteration, the algorithm calculates the gradient of the loss function with respect to the predicted values of the model, and then fits a weak learner to the negative gradient (i.e., the residual errors) of the current model. The new weak learner is then added to the model with a small weight, and the process is repeated until the desired level of accuracy is achieved.

By adding weak learners in this way, the model is able to gradually improve its performance over time, with each new learner focusing on the areas where the previous learners have made mistakes. The final model is a weighted sum of all the weak learners, with the weights determined by the boosting algorithm

Q6. How does Gradient Boosting algorithm build an ensemble of weak learners?

Ans.The Gradient Boosting algorithm builds an ensemble of weak learners by iteratively adding new learners to the model, with each new learner trying to improve the predictions of the current model.

Q7. What are the steps involved in constructing the mathematical intuition of Gradient Boosting algorithm?

Ans.

1. create base model
2. compute residual and errors
3. construct decision tree
4. repeat the step2 and so on