**Predicting the response rate on Stack Overflow with Google BigQuery:**

**Machine Learning**

**Introduction**

Stack Overflow is the leading web community for getting answers to computer programming and other technology questions. The wrangled dataset containing ~65k rows is ready to be further analyzed. Exploratory data analysis has already been completed and the data set is ready for machine learning algorithms to be applied for predictive analysis. In this project, we’re attempting to predict the time it would take for an answer to be posted for a given question.

**Handling categorical values**

The categorical features, 'first\_word', 'tag', 'weekday\_utc', 'question\_length', were handled by applying one hot encoding. This was carried out by creating dummy variables. This resulted in a dataset with 55345 rows and 27 columns.

**Machine Learning Models**

The dataset was split into a 80% train and 20% test set. For this regression analysis, we focus on tree-based Machine Learning algorithms as the features are not on the same scale. Ten-fold cross-validation was carried out for each model. The model accuracy was measured by calculating the root mean squared error (RMSE). The algorithms and the RMSE for each algorithm are listed below:

|  |  |
| --- | --- |
| **Algorithm** | **RMSE (hour)** |
| Linear Regression | 1.586 |
| Decision Tree w/o max leaf | 2.295 |
| Decision Tree w/ max leaf=250 | 1.601 |
| Random Forest | 1.590 |
| Adaboosting | 1.691 |
| Gradient Boosting | 1.569 |
| Stochastic Gradient Boosting | 1.574 |
| Extreme Gradient Boosting | 1.935 |
| Extreme Gradient Boost w/optimized DF | 1.569 |

**Hyperparameter Tuning**

Hyperparameters of the was tuned by using the GridSearch validation. The following parameter set was checked for the optimal parameter set:

* Random Forest Model:

'n\_estimators': [100, 200, 300],

'max\_features': ['log2', 'auto', 'sqrt'],

'min\_samples\_leaf': [10, 30, 50]

Best hyperparameters: {'max\_features': 'auto', 'min\_samples\_leaf': 50, 'n\_estimators': 300}

* Gradient Boosting:

'max\_depth': [2, 4, 8],

'n\_estimators': [100, 200, 300]

Best hyperparameters: {'max\_depth': 2, 'n\_estimators': 200}

**Conclusion**

Therefore, the Gradient Boosting algorithm was selected as the best model for this project. By analyzing the importance of features in this model, the length and tag of the question seem to be most important.

A picture containing screenshot

Description automatically generated