Winter Of Code 8.0 Report

Author: Mayuresh Bharti

Admission no.: 25JE1062

Branch: Electronics and Communication Engineering

Kaggle Competition:

Data Analysis:

Train Dataset:

- Number of Rows: 2000 (Excluding Header)
- Number of Columns: 3074 (ID+ Features +Y labels)
- Distribution of Classes: 0's=1800 and 1's=200 (Highly Unbalanced)

Test Dataset:

- Number of Rows: 2000
- Number of Columns: 3073(ID + Features)

Models:

1. Started with the Basic Linear Regression model from sklearn library with accuracy as the measure. Just trained a raw model without splitting the training dataset. Public leaderboard score: 0.82
2. Since a linear model might not be suitable for such an unbalanced dataset I switched to more complex models. Starting with SVC Classifier combined with GridSearchCV and Pipeline. It boosted the public leaderboard score to 0.99 but was highly biased towards predicting 0 (majority class dominating the learning leading to overfitting). Made some improvements on this model by including parameters like class_weights. This model performed much better but the positive prediction rate was too high (approximately 900 1's as compared to 200 1's in the train dataset)
3. Next made a Decision Tree Ensemble with the XGBoost Classifier. Included scale_pos_weight to handle class imbalance and used cross_val_score to calculate f1 scores. Then used balanced accuracy to tune Threshold (which was by default 0.5). Also used l1 and l2 regularization. (One of two final submissions was from this model)

4. **Also made a Logistic Regression model from sklearn but this time setting class_weights as balanced, l1 penalty and saga solver. Then did Threshold tuning using cv f1 score. Although the scores of this model were good but the positive prediction rate was too high so again did threshold tuning by checking the positive prediction rate by each threshold. (The other submission was from this model).**

MACHINE LEARNING LIBRARY

Linear Regression:

Data Analysis:

Train Dataset:

- **Number of Rows: 600000 (Excluding Header)**
- **Number of Columns: 16 (Features +Y labels)**

Test Dataset:

- **Number of Rows: 150000**
- **Number of Columns: 15(Features)**

Functions and implementations:

- **Includes functions such as comp_cost, comp_gradient and gradient descent for learning and evaluating.**
- **The Pipeline for training goes like this:**

Extracting the dataset

Filling null columns with median values

Splitting the dataset into train, cross validation and test sets

Performing z score normalization on all three datasets using mean and standard deviation of the train dataset

Creating grid for regularization parameter and running for loop to find the best one based on the lowest cv loss

Final training using the best regularization parameter and then checking the r2

score on the test dataset

Results:

Best regularization parameter (lambda) found=0.1

Final Test R^2 Score: 0.968

Polynomial Regression:

Data Analysis:

Train Dataset:

- **Number of Rows: 1200 (Excluding Header)**
- **Number of Columns: 8 (Features +Y labels)**

Test Dataset:

- **Number of Rows: 400**
- **Number of Columns: 7(Features)**

Functions and implementations:

- **Includes functions such as comp_cost, comp_gradient and gradient descent for learning and evaluating.**
- **The Pipeline for training goes like this:**

Extracting the dataset

Filling null columns with median values

Splitting the dataset into train, cross validation and test sets

Performing z score normalization on all three datasets using mean and standard deviation of the train dataset

Creating grid for regularization parameter and running for loop to find the best one based on the lowest cv loss

Final training using the best regularization parameter and then checking the r2 score on the test dataset

Logistic Regression (Binary Classification):

Data Analysis:

Train Dataset:

- **Number of Rows: 50000 (Excluding Header)**
- **Number of Columns: 26 (Features +Y labels)**
- **Targets: 1 or 0**

Test Dataset:

- **Number of Rows: 25000**
- **Number of Columns: 25 (Features)**

Functions and implementations:

- **Includes same functions such as comp_cost_logistic, comp_gradient_logistic and gradient descent for learning and evaluating the model using sigmoid function and Binary Cross Entropy loss.**
- **The Pipeline for training goes like this:**

Extracting the dataset

Filling null columns with median values

Splitting the dataset into train, cross validation and test sets

Performing z score normalization on all three datasets using mean and standard deviation of the train dataset

Creating grid for regularization parameter and running for loop to find the best one based on the highest cv f1 score

Final training using the best regularization parameter and then checking the f1 score on the test dataset

Results:

Best regularization parameter (lambda) found=0

Final Test R^2 Score: 0.9995

Logistic Regression (Multiclassification)

Data Analysis:

Train Dataset:

- **Number of Rows: 50001 (Excluding Header)**
- **Number of Columns: 41 (Features +Y labels)**
- **Targets: 0, 1, 2, 3, 4**

Test Dataset:

- **Number of Rows: 25000**
- **Number of Columns: 40 (Features)**

Functions and implementations:

- **Includes same functions such as comp_cost_softmax, comp_gradient_softmax and gradient descent for learning and evaluating the model but this time using so softmax function and Categoriacal Cross Entropy loss.**
- **Some additional functions for computing class weights (in case of unbalanced data) and making one hot encodings of the y labels**
- **The Pipeline for training goes like this:**

Extracting the dataset

Filling null columns with median values

Splitting the dataset into train, cross validation and test sets

Performing z score normalization on all three datasets using mean and standard deviation of the train dataset

Creating grid for regularization parameter and running for loop to find the best one based on the highest cv f1 score

Final training using the best regularization parameter and then checking the f1 score on the test dataset