**AML documentation**

**Overview:**

**This document presents three distinct machine learning models developed for different purposes. Each model is trained on specific datasets and employs various algorithms tailored to the nature of the data and the intended predictions.**

**Datasets:**

**1) Codeforces Dataset:**

**This dataset consists of two separate DataFrames extracted from Codeforces, a prominent competitive programming platform. It aims to facilitate research and analysis in machine learning by providing insights into Codeforces user ratings and participation.**

**https://www.kaggle.com/datasets/meruvulikith/codeforces-ratings-classification-and-regression?select=classification\_df.csv**

**Classification DataFrame:**

**Features:**

**userid: Unique identifier for each Codeforces user.**

**rank-type: Indicates the skill level of users, categorized into various ranks.**

**contest1 to contest10: Performance ratings of users in ten different Codeforces contests.**

**Regression DataFrame:**

**Features:**

**userid: Unique identifier for Codeforces users.**

**rating: The maximum rating achieved by users on Codeforces.**

**contest1 to contest10: Performance ratings of users in ten different Codeforces contests.**

**Total Samples: 1423**

**2) Employee Dataset:**

**This dataset is provided by a company's HR department with the goal of predicting whether certain employees would leave the company within the next two years.**

**Features:**

**Education: Level of education.**

**JoiningYear: Year of joining the company.**

**City: Office location where the employee is posted.**

**PaymentTier: Payment tier (1: highest, 2: mid-level, 3: lowest).**

**Age: Current age of the employee.**

**Gender: Gender of the employee.**

**EverBenched: Whether the employee has been kept out of projects for one month or more.**

**ExperienceInCurrentDomain: Experience in the current field.**

**LeaveOrNot: Binary variable indicating whether the employee leaves the company.**

**Machine Learning Models:**

**https://www.kaggle.com/datasets/tejashvi14/employee-future-prediction**

**1) Support Vector Machine Model:**

**Dataset Used: Codeforces Dataset (Regression)**

**Description: This regression model utilizes the scores from ten Codeforces contests to predict the rank of users.**

**Total Samples: 1423**

**Split: 30% testing (427 samples) and 70% training (996 samples).**

**support vector we use liner kearnl.**

**2) Classification Tree:**

**Dataset Used: Employee Dataset**

**Description: Employing a Decision Tree Classifier with alpha=0.00086, this classification model predicts whether employees will leave the company based on various features.**

**Total Samples: 4654**

**Split: 33% testing (1535 samples) and 67% training (3118 samples).**

**3) Artificial Neural Network:**

**Dataset Used: Employee Dataset**

**Description: Utilizing features such as education level, joining year, city, payment tier, age, gender, ever benched status, and experience in the current domain, this neural network model predicts whether employees will leave the company.**

**Total Samples: 4654**

**Split: 20% testing (930 samples) and 80% training (3723 samples).**

**ANN:**

**The extracted features are:**

**Categorical features: 'Education', 'City', 'Gender', 'EverBenched'**

**Numerical features: 'JoiningYear', 'PaymentTier', 'Age', 'ExperienceInCurrentDomain'**

**The dimension of the resulting features after encoding and scaling will be the same as the original dataset, with the addition of one-hot encoding for categorical features and normalization for numerical features.**

**Cross-validation:**

**Cross-validation is not explicitly used in the provided code snippet for model training.**

**Hyperparameters:**

**Initial Learning Rate: Not explicitly specified in the provided code. The default learning rate for the Adam optimizer is typically used, which is 0.001.**

**Optimizer: Adam optimizer (optimizer='adam') is used.**

**Regularization: Dropout regularization (model.add(Dropout(0.05))) is applied after the first dense layer with a dropout rate of 0.05.**

**Batch Size: 32 (batch\_size=32) is used.**

**Number of Epochs: 50 (epochs=50) is used for training.**

**Results Details:**

**Loss Curve: Plot of training and validation loss over epochs.**

**Accuracy: Accuracy achieved on the testing dataset.**

**Confusion Matrix: Visualization of the confusion matrix to evaluate model performance.**

**A graph showing a line that is blue

Description automatically generated with medium confidenceA diagram of a confusion matrix

Description automatically generated**

**Confusin Matrix is :**

**[[587 23]**

**[102 219]]**

**#####################################################**

**accuracy is :**

**0.8657357679914071**

**#####################################################**

**recall is :**

**0.6822429906542056**

**#####################################################**

**precision is :**

**0.9049586776859504**

**#####################################################**

**f1 score is :**

**0.7779751332149201**

**Classification Tree:**

**Feature Extraction Phase:**

**Number of features : 7**

**Cross-validation:**

**Cross-validation is not implemented in the provided code.**

**Hyperparameters:**

**In the provided code, the Decision Tree Classifier hyperparameter ccp\_alpha is set to 0.00086.**

**Other hyperparameters such as initial learning rate, optimizer, regularization, batch size, and number of epochs are not applicable to the Decision Tree Classifier. They are typically used in deep learning models like neural networks.**

**Results Details:**

**Loss Curve: Not applicable for Decision Tree Classifier.**

**Accuracy: Accuracy achieved on the testing dataset.**

**Confusion Matrix: Visualization of the confusion matrix to evaluate model performance.**

**ROC Curve & AUC: Not applicable for Decision Tree Classifier.**

**A line graph with a point in the center

Description automatically generated with medium confidenceA chart with numbers and labels

Description automatically generated with medium confidence**

**Training score: 0.87**

**Testing score: 0.85**

**------------------------------------------------------------**

**The accuracy score: 0.85**

**The f1-score: 0.73**

**SVM:**

**Feature Extraction Phase:**

**Number of features : 10**

**Cross-validation:**

**Cross-validation is not explicitly used in the provided code. If you wish to incorporate it, you can use sklearn.model\_selection.cross\_val\_score() or related functions.**

**Specify the number of folds and the ratio of training/validation within each fold.**

**Hyperparameters:**

**The SVR model is configured with the linear kernel.**

**Other hyperparameters such as C, epsilon, and gamma (for non-linear kernels) can be tuned for better performance.**

**Results Details:**

**The provided code computes the Mean Absolute Error (mean\_absolute\_error) and R^2 Score (r2\_score) for evaluating the model's performance on the testing data.**

**Additionally, there's a function predict\_SVR(ranks) provided for making predictions using the trained SVR model.**

**Visualization (scatter plot) comparing actual vs. predicted values can also be uncommented for further analysis.**

**For a more comprehensive evaluation, you may consider computing additional metrics like Mean Squared Error, Root Mean Squared Error, etc.**

**A graph with orange dots

Description automatically generated**

**User**

**SVM:**

**Accuarcy:**

**R^2 Score: 87%**

**Mean Absolute Error: 55.55618974805897**