Journal of Cloud Computing:
Advances, Systems and Applications

RESEARCH

Open Access

Predicting the total Unified Parkinson's Disease Rating Scale (UPDRS) based on ML techniques and cloud-based update

Sahand Hamzehei[1], Omid Akbarzadeh[1], Hani Attar[2], Khosro Rezaee[3*], Nazanin Fasihihour[1] and Mohammad R. Khosravi[4]

Abstract

Nowadays, smart health technologies are used in different life and environmental areas, such as smart life, healthcare, cognitive smart cities, and social systems. Intelligent, reliable, and ubiquitous healthcare systems are a part of the modern developing technology that should be more seriously considered. Data collection through different ways, such as the Internet of things (IoT)-assisted sensors, enables physicians to predict, prevent and treat diseases. Machine Learning (ML) algorithms may lead to higher accuracy in medical diagnosis/prognosis based on health data provided by the sensors to help physicians in tracking symptom significance and treatment steps. In this study, we applied four ML methods to the data on Parkinson's disease to assess the methods' performance and identify the essential features that may be used to predict the total Unified Parkinson's disease Rating Scale (UPDRS). Since accessibility and high-performance decision-making are so vital for updating physicians and supporting IoT nodes (e.g., wearable sensors), all the data is stored, updated as rule-based, and protected in the cloud. Moreover, by assigning more computational equipment and memory in use, cloud computing makes it possible to reduce the time complexity of the training phase of ML algorithms in the cases we want to create a complete structure of cloud/edge architecture. In this situation, it is possible to investigate the approaches with varying iterations without concern for system configuration, temporal complexity, and real-time performance. Analyzing the coefficient of determination and Mean Square Error (MSE) reveals that the outcomes of the applied methods are mostly at an acceptable performance level. Moreover, the algorithm's estimated weight indicates that Motor UPDRS is the most significant predictor of Total UPDRS.

Keywords Machine learning, Lineal regression, Linear least square, Conjugate gradient, Adam optimization algorithm, Ridge regression, Machine learning for healthcare, Parkinson's disease

Introduction

Numerous industries now profit from cutting-edge technology by disseminating technology to the populace. A recent study demonstrates the engagement of several researchers in wireless communication [1–5] to improve an existing system by addressing pertinent difficulties. Some areas of research, such as AI, have played a crucial part in the evolution of intelligence over these years to develop different projects in areas such as image recognization [6]. As stated by [7], remote-controlled robots will soon become more widespread in various fields. ML, which utilizes historical data as input and can predict the

*Correspondence:
Khosro Rezaee
kh.rezaee@meybod.ac.ir
[1] Department of Electronics and Telecommunications, Polytechnic University of Turin, Turin, Italy
[2] Department of Energy Engineering, Zarqa University, Zarqa, Jordan
[3] Department of Biomedical Engineering, Meybod University, Meybod, Iran
[4] Weifang University of Science and Technology, Weifang, Shandong, China

future based on the output [8], is a discipline that handles a wide range of problems in several fields [9]. The importance of ML may be observed in several areas of health care, including genomic medicine [10], cancer detection [11], and early diabetes diagnosis [12]. Moreover, the integration of ML and other technologies, such as the IoT, resulted in the smart hospital development project developed by [13, 14] in order to manage hospitalized patients efficiently. The other examples are [15, 16] developed an intelligent system for discovering social distance in the hospital during the pandemic. The last example compares ML methods to classified bioinformatics data [17]. This research concentrates on one of the diseases known as Parkinson's disease (PD), a chronic degenerative disorder of the Central Nervous System (CNS) that predominantly affects the motor system. According to [18], PD is the most quickly spreading neurological disorder globally. Marras et al. [19] forecasts that by 2030, the number of Americans with PD will increase to 1.2 million from the current one million. Analyzing, understanding, and predicting the sign of PD is essential as, according to [20], there is no cure for PD, and the only treatment options are medicines, lifestyle modifications, and surgery. As there is no cure for PD at present, the Unified Parkinson's disease Rating Scale (UPDRS) is used to monitor the course of the condition. This paper aims to apply different ML methods to understand their performance based on the MSE and R-Squared (R^2). Figure 1 represents summarizing steps, from collecting data to determining the model's evaluation of the different algorithms. We proposed four linear regression methods, outlined in Table 1, for application to the PD data and evaluation to determine the most accurate.

Table 1 Applied linear regression methods

Linear Least Square (LLS)
Conjugate Gradient
Adam optimization algorithm
Ridge Regression

Prerequisites

Data Collection

The dataset used in the study reaches from the UC Irvine ML repository, which was created by Athanasios Tsanas and Max Little of the University of Oxford [21]. Table 2 represents the columns of the available dataset after dropping test time (since it is not considered in this paper).

Dataset consists of 5876 rows and 21 columns. Each patient is related to one row in the dataset, characterized by twenty features (after dropping test time). Each patient's information is gathered in an ordered manner. On a similar day of medical examination, it is feasible for the same patient to have many rows with the same UPDRS result but distinct values for Shimmer, Jitter, etc. Various voice recordings have been analyzed to determine the values. The dataset rows should be shuffled to ensure the model stays general and has lower overfit. Shuffling the rows causes improvement in the ML model quality and predictive performance. Then the data is split into three different parts,

- Training, 50% of the original dataset.
- Validation, 25% of the original dataset.
- Test, 25% of the original dataset.

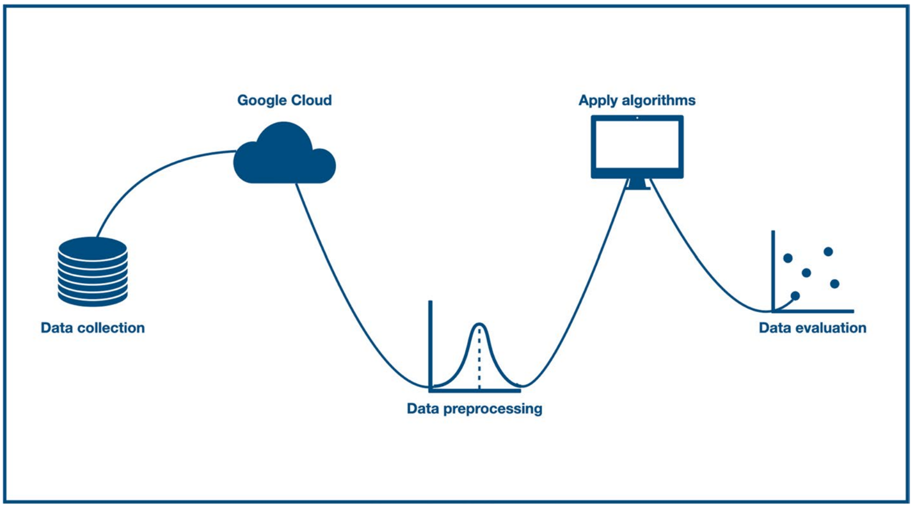


Fig. 1 Summarizing the steps of the paper

Table 2 Dataset columns (features)

Age	Shimmer (dB)
Sex	Shimmer: 3-point Amplitude Perturbation Quotient (APQ3)
Motor UPDRS	Shimmer: 5-point Amplitude Perturbation Quotient (APQ5)
Total UPDRS	Shimmer: 11-point Amplitude Perturbation Quotient (APQ11)
Jitter (%)	Shimmer: Average Absolute Differences (DDA)
Jitter absolute (Abs) in ms	Noise-to-Harmonics Ratio (NHR)
Jitter: Average Relative Perturbation (RAP)	Harmonics-to-Noise Ratio (HNR)
Jitter: 5-point Period Perturbation Quotient (PPQ5)	Recurrence Period Density Entropy Measure (RPDE)
Jitter: Difference of Differences of Periods (DDP)	Signal fractal scaling exponent of Detrended Fluctuation Analysis (DFA)
Shimmer	Pitch Period Entropy (PPE)

Because of the different range of features and to improve the efficiency of the algorithms, data should normalize by applying the normalization of the data concerning the training dataset by using Eqs. 1, 2, and 3.

$$z\_train\ norm = \frac{X\_train - \mu\_train}{\sigma\_train} \tag{1}$$

$$z\_validation\ norm = \frac{X\_val - \mu\_train}{\sigma\_train} \tag{2}$$

$$z\_test\ norm = \frac{X\_test - \mu\_train}{\sigma\_train} \tag{3}$$

Where,

- $z\_train\ norm$ is the normalized training set.
- $z\_validation\ norm$ is the normalized validation set.
- $z\_test\ norm$ is the normalized test set.
- $\mu\ train$ is the mean row vector of the training set.
- $\sigma\ train$ is the standard deviation row vector of the training set.

The normalization changes the values of numeric columns in the dataset into a standard scale without distorting differences in the ranges of values. Now, by considering the correlation, the linear relationship between two variables is determined. A strong correlation is called if the coefficient value is between + 0.50 and +1. Moderate correlation is used while the value is between +0.30 and +0.49. A value below +0.29 is called a low correlation. Figure 2 shows the covariance matrix after dropping irrelevant features. From Fig. 2, it can be seen that there is a strong correlation between motor UPDRS and total UPDRS; shimmer and jitter parameters have a moderate correlation with each other. In contrast, there is a low correlation between all other parameters and total UPDRS. Only highly and moderately correlated features have been evaluated to assess the final accuracy of the Machine Learning algorithms. However, it can be seen that in medical datasets, features may occasionally be retrieved after checking with the physician to determine the value of features in the actual world.

Cloud-Based Computing Services and Updates

One of the main problems related to ML algorithms is storing, processing, and updating the data efficiently. However, the traditional way, the local host, is still popular among researchers and developers [22, 23]. One solution to locally distributed host issues is to benefit from the concept of cloud services, which enables developers and researchers to access applications and data remotely [24, 25] in a platform with a huge amount of storage capacity and computational power. Storing and processing data in Cloud services provides benefits, including,

- Increasing the security of code and data and protecting them against hacker attacks.
- Providing accessibility. In this case, accessing the data anytime from anywhere is possible.
- Allocating extra computational resources, which may result in reducing the time complexity of the processing. The cloud service enables developers to increase their code privacy and flexibility and, in advanced cases, reduce the system's time complexity and power consumption.

On the other hand. cloud computing can facilitate the fast and secure processing of distributed end-user data. As an illustration, smart sensors collect users' medical information, such as blood pressure and heart rate, then send those to the cloud for further processing, where the machine learning algorithms operate. Using the cloud-based updates, we can periodically refresh the training data to make more precise predictions and decisions. Cloud-based control also allows us to involve a human expert in the case of sensitive situations, such as the timely prediction of a heart attack. In particular, this

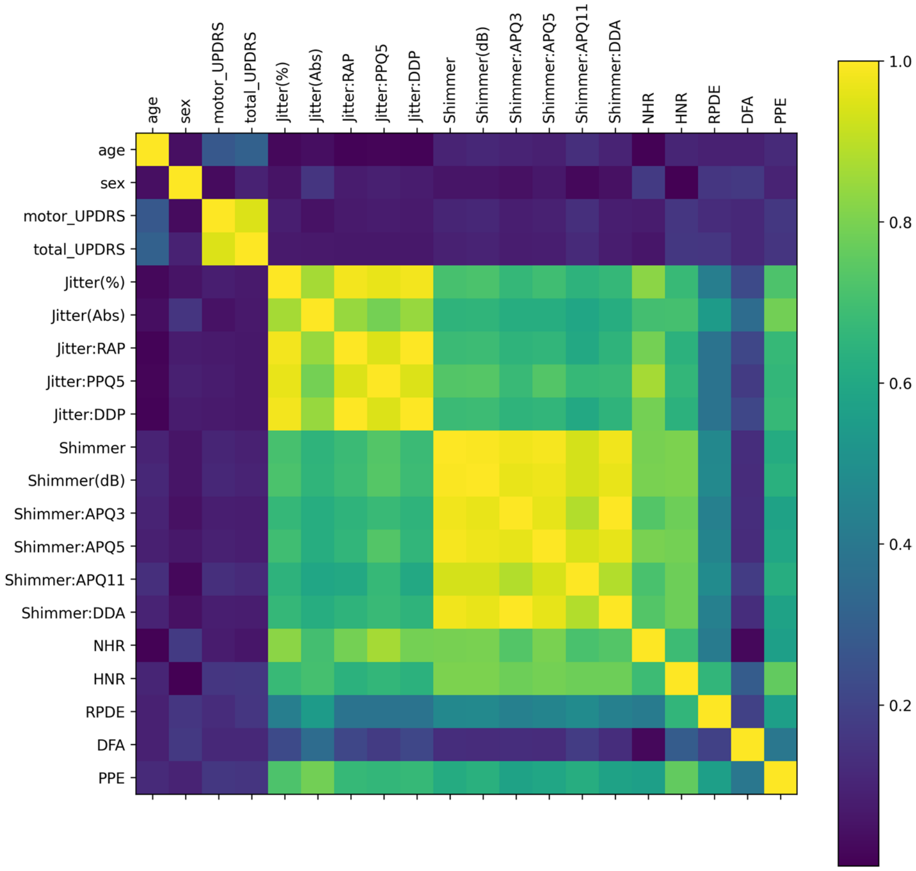


Fig. 2 The covariance matrix of features

paper has used Google Colab, a free cloud service that enables the execution of Python code in Jupyter Notebook format [26, 27]. It is an additional use of the cloud in addition to the justification provided above. Moreover, it is possible to use Google Cloud ML APIs for further processing by considering its cloud facilitations. Figure 3

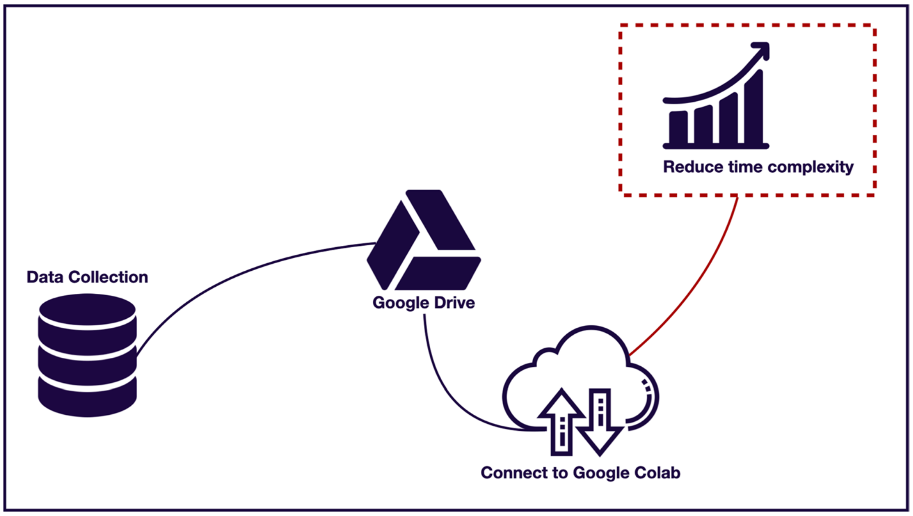


Fig. 3 Steps to reduce the time complexity of training ML methods

represents how it is possible to achieve better time complexity by using the concept of the Cloud.

Materials and methods

Linear Regression

In a vector $Y = f(X), Y$ is a random variable, which depends on other random variables $X_1, X_2, X_3, ..., X_F$. Y is a random scalar variable, X is a random vector variable, and $f()$ is an unknown function [10]. By measuring X and knowing the function $f()$, it is feasible to predict Y. This prediction may be advantageous for the direct measurement of Y because measuring X is less costly, less invasive, less harmful to the patient, and permits many more estimations per day. To find $f()$, the values of Y and X should be measured for N times as expressed in Eq. 4.

$$y(n) = f(x(n)),\ n = 1, ..., N \tag{4}$$

Where y represents a measured value and x determines a vector of numbers. By assuming observed values $y(n)$, $n = 1, ..., N$, $y(n) \in R$, and back to the vector of variables $x(n)$, $(x(n) \in R^F)$, it is possible to predict the future values of $y(n)$, by using feature values of $x(n)$. $x(n)$, called the "regressor", is the vector of the independent variable. In contrast, $y(n)$, known as the "regressand", is the dependent variable; the relationship between them is unknown, as shown in Fig. 4.

In this paper, index $n \in [1, N]$ identifies the patient, and $f \in [1, F]$ specifies the features so that $x_f(n)$ is the $f - th$ feature of the $n - th$ patient. In linear regression, the assumption is represented in Eq. 5.

$$Y = w_1 X_1 + w_2 X_2 + ... + w_F X_F \tag{5}$$

Which measured values are all affected by errors. Equation 6 denotes the vector of the "regressors".

$$y(n) = [x(n)]^T w + v(n) \tag{6}$$

Where the vector of the "regressors" $x(n)$ and w are the set of weights to be found expressed in Eqs. 7 and 8, respectively. The weight of a feature indicates how significant it is for the following groups of methods. This shows that any particular approach it may be altered dependent on the algorithm that is being studied. Furthermore, $v(n)$ is the measurement error.

$$x(n) = [x_1(n), ..., x_F(n)]^T \tag{7}$$

$$w = [w_1, ..., w_F] \tag{8}$$

To expand the problem for several patients, it is possible to write the vector as a matrix, in which each row represents an individual patient, as argued in Eq. 9.

$$\begin{bmatrix} y(1) \\ y(2) \\ . \\ . \\ . \\ y(N) \end{bmatrix} = \begin{bmatrix} x_1(1) & x_2(1) & x_3(1) & ... & x_F(1) \\ x_1(2) & x_2(2) & x_3(2) & ... & x_F(2) \\ . & . & . & ... & . \\ . & . & . & ... & . \\ . & . & . & ... & . \\ x_1(N) & x_2(N) & x_3(N) & ... & x_F(N) \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ . \\ . \\ . \\ w_F \end{bmatrix} \begin{bmatrix} v(1) \\ v(2) \\ . \\ . \\ . \\ v(N) \end{bmatrix} \tag{9}$$

This paper applies the different linear regression models mentioned in Table 2 to the provided dataset to estimate UPDRS as a linear regression problem.

LLS

Finding the straight line across a group of data points that provides the greatest possible fit is the objective of the LLS approach, which is both the simplest and most often used type of linear regression [10]. Considering LLS makes it possible to write the vector of measurement Y as Eq. 10.

$$Y = X^T.w + v(n) \tag{10}$$

Where,

- X has dimensions, number of patients, and features.
- w is a column vector of weights, and it is unknown.
- Y the experimental target data.
- $v(n)$ is a column vector of errors.

w is the unknown vector, and to find out that, the concept of square error is considered, which is the function of w, as demonstrated in Eq. 11.

$$f(w) = ||y - X.w||^2 \tag{11}$$

Fig. 4 Unknown relationship between "regressor" and "regressand"

Equations 12 and 13 represent how to find the minimum of the function, the gradient of *f*(*w*) should be evaluated and then set equal to 0, and at the end, *w* can be found.

$$\nabla f(w) = -2.X^T.y + 2.X^T.X.w = 0 \tag{12}$$

$$w = (X^T.X)^{-1}.X^T.y \tag{13}$$

Once the optimal value of vector w has been determined, this estimate may be inserted into the *f*(*w*) formula, and the minimum is then calculated. Figures 5, 6, 7, and 8 show the results obtained in LLS. From Fig. 5, it can be seen that, by considering LLS, Jitter RAP and Shimmer DDA have a more significant weight than other features. High values of Jitter and Shimmer usually confirm high instability during sustained vowel production. Figures 6 and 7 show the results of the comparison of estimation *y* and true values for training and test datasets, respectively. The similarity between training and testing appears by comparing them, which describes models that

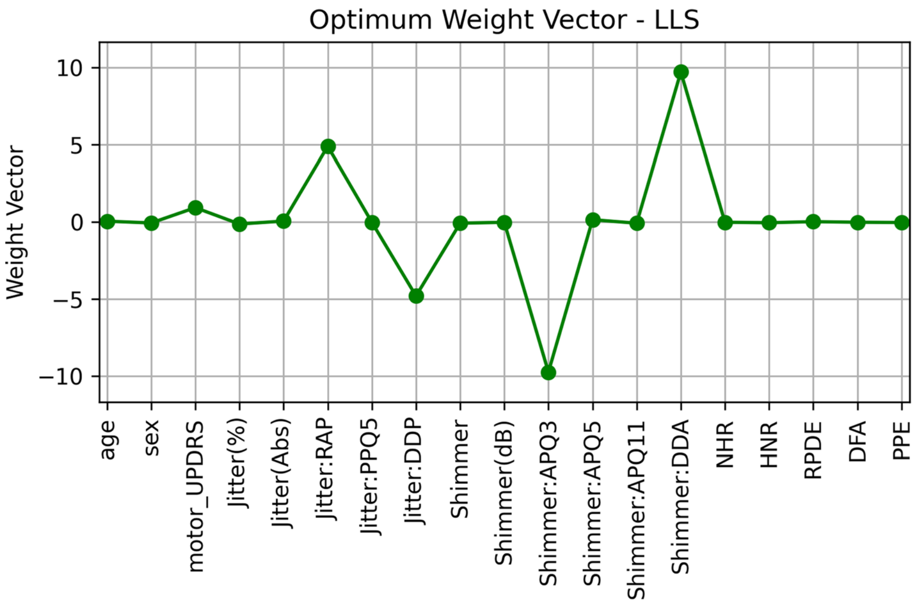


Fig. 5 Optimized weights for LLS

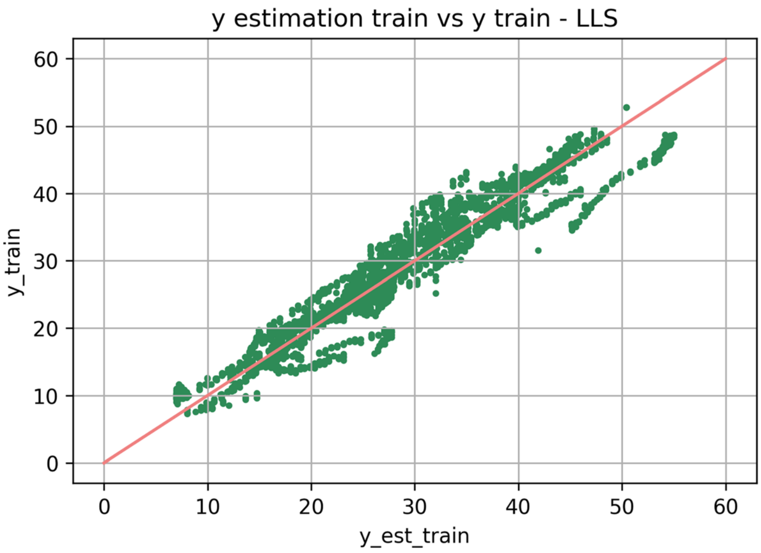


Fig. 6 y estimation train vs y train

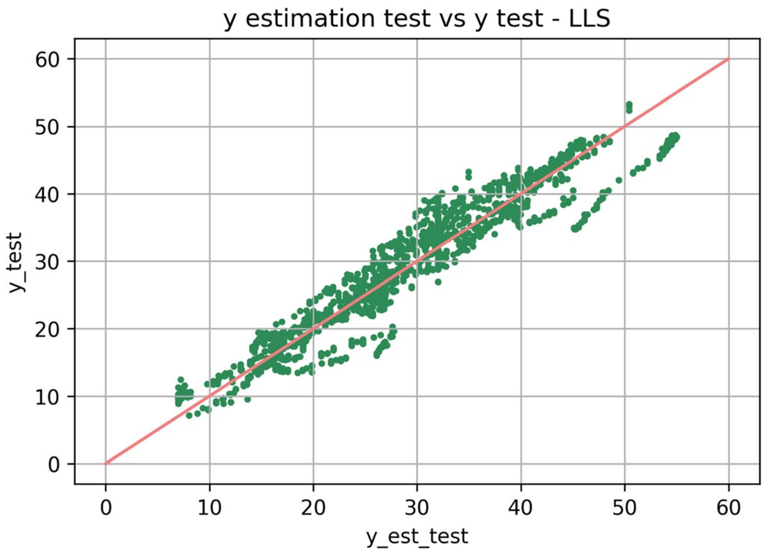


Fig. 7 y estimation test vs y test

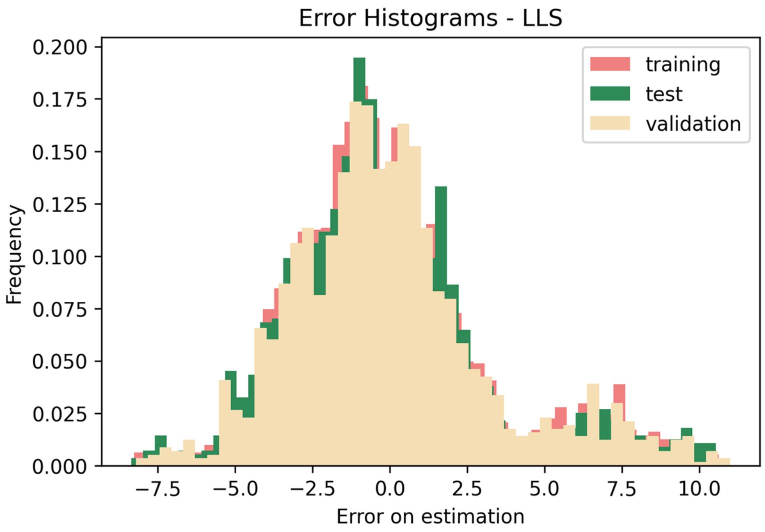


Fig. 8 Error histogram

generalize successfully. Figure 8 also shows the similarities between the estimated and true values.

Conjugate Gradient

The Conjugate Gradient technique is generally used as an iterative algorithm; however, it can be used as a direct method and produce a numerical solution [28]. The assumption is that two vectors of d_i and d_k are conjugate concerning the symmetric matrix of Q if Eq. 14 is true.

$$d_i^T Q d_k = 0 \tag{14}$$

To generate conjugate vectors, one of the solutions is considering Eq. 15.

$$Qu_k = \lambda_k u_k \tag{15}$$

Where Q is symmetric, and therefore $U^{-1} = U^T$ and the eigenvectors are orthogonal. In this case, the eigenvectors are also Q-orthogonal. The conjugate gradient algorithm starts from the initial solution, which means $w_0 = 0$. It evaluates the function's gradient as stated in Eq. 16.

$$g_0 = \nabla f(w_0) = Qw_0 - b = -b \tag{16}$$

Approximation of w can be found through Eq. 17.

$$w_1 = w_0 + \alpha_0 d_0 \tag{17}$$

Where α_0 can be obtained by using Eq. 18

$$\alpha_0 = \frac{d_0^T b}{d_0^T Q d_0} \tag{18}$$

The conjugate gradient algorithm is defined through Eqs. 19, 20, 21, 22, and 23 to solve the system $Qwb = 0$, where Q is symmetric and positive definite in $\mathbb{R}^{N*N}$. The algorithm is started by setting $d_0 = -g_0 = b$, initial solution as $w_0 = 0$, and $k = 0$.

$$\alpha_k = -\frac{d_k^T g_k}{d_k^T Q d_k} \tag{19}$$

$$w_{k+1} = w_k + \alpha_k d_k \tag{20}$$

$$g_{k+1} = Qw_{k+1} - b = g_k + \alpha_k Q d_k \tag{21}$$

$$\beta_k = \frac{g_{k+1}^T Q d_k}{d_{k+1}^T Q d_k} \tag{22}$$

$$d_{k+1} = -g_{k+1} + \beta_k d_k \tag{23}$$

Defining the stopping condition of N makes it possible to understand when the algorithm is stopped. If k reaches the N, the threshold is met. Otherwise, the procedure is repeated. Figures 9, 10, 11, and 12 illustrate the results obtained by applying the Gradient Algorithm. The solution has been applied using 100000 iterations and setting the learning coefficient 10^5. Figures 10, 11, and 12 can be seen as the acceptable performance of the conjugate method. By comparing Figs. 10 and 11 with Figs. 6 and 7, it can be seen that both conjugate and LSS have almost the same performance in the case of generalization. However, the optimal weight vector has a different story. As seen in Fig. 5, shimmer DDA reaches ten value, but this number for Conjugate, Fig. 9, is 0. Among all the features, motor UPDRS stands first and has the highest weight value.

Adam optimization algorithm

Adam stands for Adaptive Moment Estimation. It is an extension of gradient descent. The term “gradient descent” refers to the first-order iterative optimization process used to locate a local minimum or maximum of a differentiable function. There are some points behind Adam’s optimization [10]. The k-th central moment of a random variable α with mean μ is defined as Eq. 24.

$$m^{(k)} = E\{(\alpha - \mu)^k\} \tag{24}$$

The variance of a random variable is the second central moment; the $k - th$ moment of a random variable α is defined in Eq. 25.

$$\mu^{(k)} = E\{\alpha^k\} \tag{25}$$

This optimization is called Adam because of using the estimation of the first and second moments of the

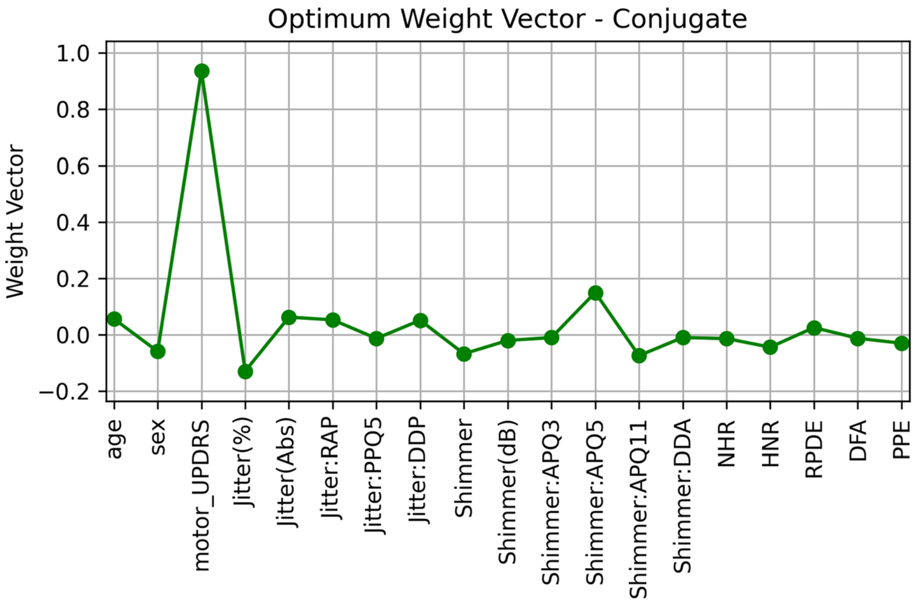


Fig. 9 Optimized weights for conjugate gradient algorithm

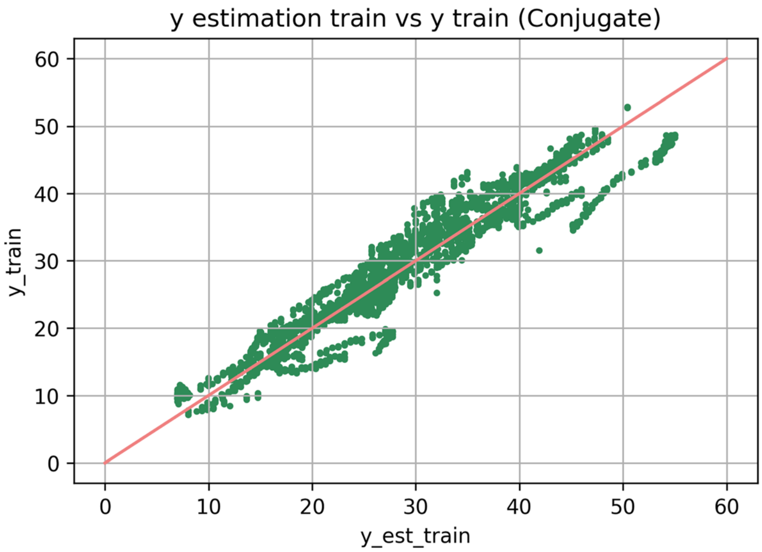


Fig. 10 y estimation train vs y train

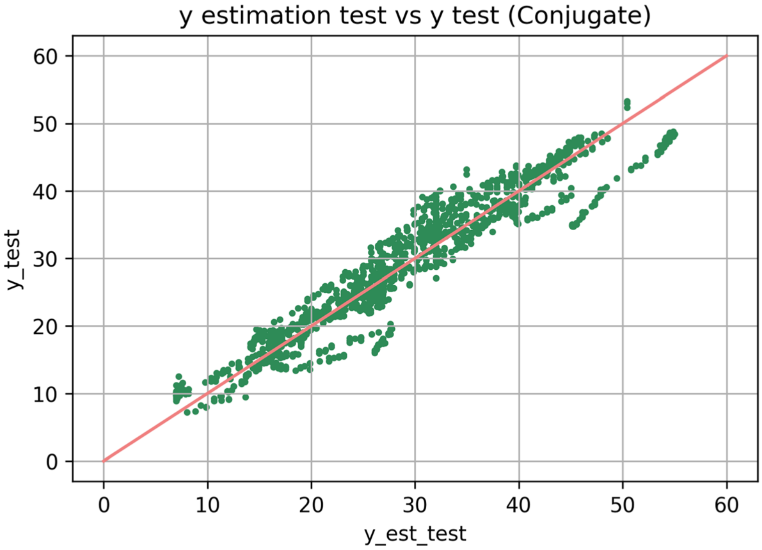


Fig. 11 y estimation test vs y test

gradient to adapt the learning rate for the weight of the neural network. Adam tests Eqs. 26 and 27.

$$\mu_i^{(1)} = m = \beta_1 \mu_{i-1}^{(1)} + (1 - \beta_1) \nabla f(x_s) \tag{26}$$

$$\mu_i^{(2)} = \nu = \beta_2 \mu_{i-1}^{(2)} + (1 - \beta_2) [\nabla f(x_s)]^2 \tag{27}$$

Where

- $\nabla f(x_s)$ is a current gradient.
- m and ν are moving average.
- β is the hyperparameter with the default values of 0.9 and 0.999.

The correction factor is introduced due to the relationship between true moments, which is estimated, as mentioned in Eqs. 28 and 29.

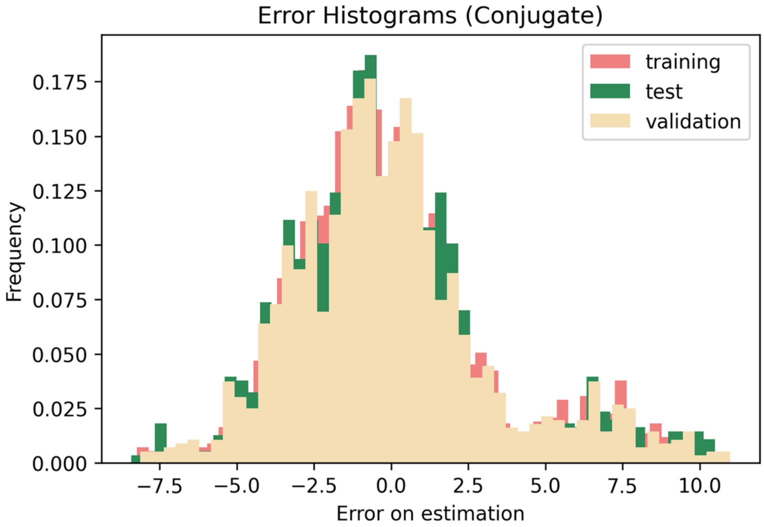


Fig. 12 Error histogram

$$\hat{m} = \hat{\mu}_i^{(1)} = \frac{\hat{\mu}_i^{(1)}}{1 - \beta_1^{i+1}} \tag{28}$$

$$\hat{v} = \hat{\mu}_i^{(2)} = \frac{\hat{\mu}_i^{(2)}}{1 - \beta_2^{i+1}} \tag{29}$$

The only step left is to use moving averages to scale the learning rate individually for each parameter. w is calculated by Eq. 30.

$$w_{i+1} = w_i - \gamma \frac{\hat{\mu}^1}{\sqrt{\hat{\mu}_i^2 + \epsilon}} \tag{30}$$

Where γ is the learning rate or step size. The proportion that weights are updated. Larger values result in faster initial learning before the rate is updated. Smaller values slow the learning rate down during training. ϵ is a tiny number to prevent any division by zero in the implementation. Also, it is used as a stopping condition parameter in the algorithm when the new minimum value of

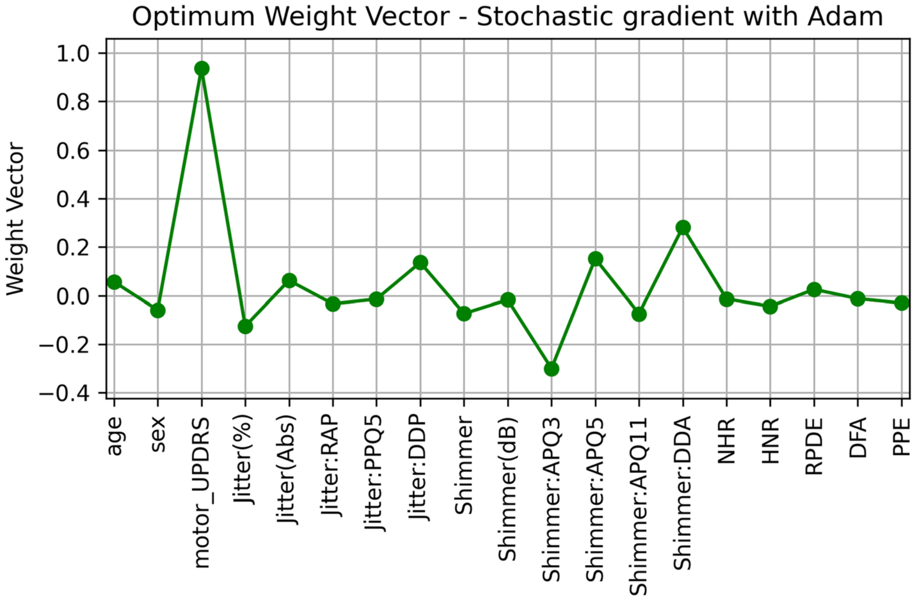


Fig. 13 Optimized weights for Stochastic gradient with Adam

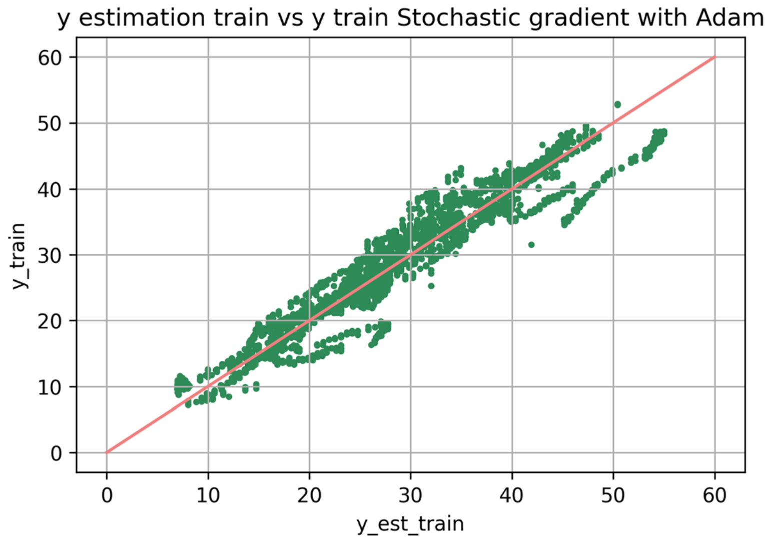


Fig. 14 y estimation train vs y train

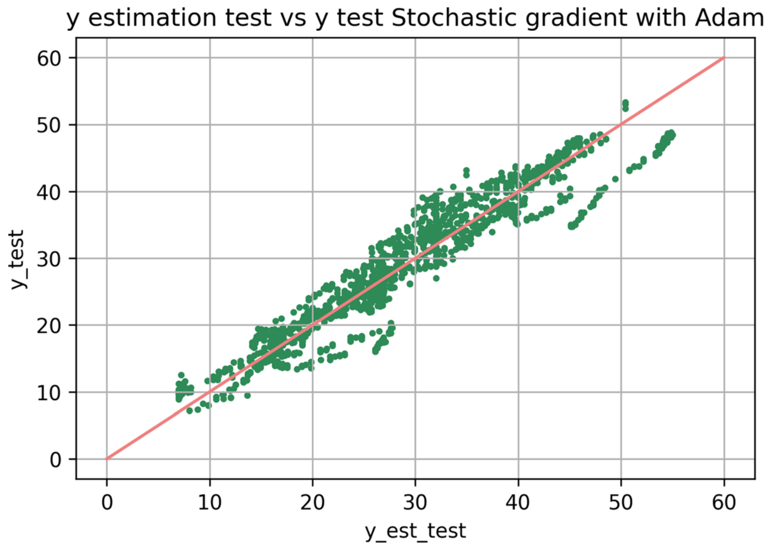


Fig. 15 y estimation test vs y test

the gradient is calculated after each iteration. It is not decreasing more than the value of ϵ. Figures 13, 14, 15, and 16 demonstrate the result obtained from the Stochastic gradient with Adam. There is a point behind the algorithm were choosing the number of iterations is very significant. If it is too small, the reduction in error will be prolonged, while if it is too large, divergent oscillations in the error plot will occur. In this paper learning coefficient and the number of iterations are set to 0.001 and 2000 after testing the result. By observing Fig. 13, the Total UPDRS depends mostly on Motor UPDRS. This is the same result that Conjugate Gradient got. As stated by Fig. 16, the error is distributed around intervals near zero, which is similar to Figs. 8 and 12. In Figs. 14 and 15, it can be observed that the data's prediction follows the axis bisector, which confirms that it is correct most of the time. Stochastic gradient with Adam in Python can be shown Fig. 17.

Ridge Regression

Multicollinearity is a significant issue with data. In multicollinearity, least squares are unbiased, and variances are substantial, resulting in predictions far from real

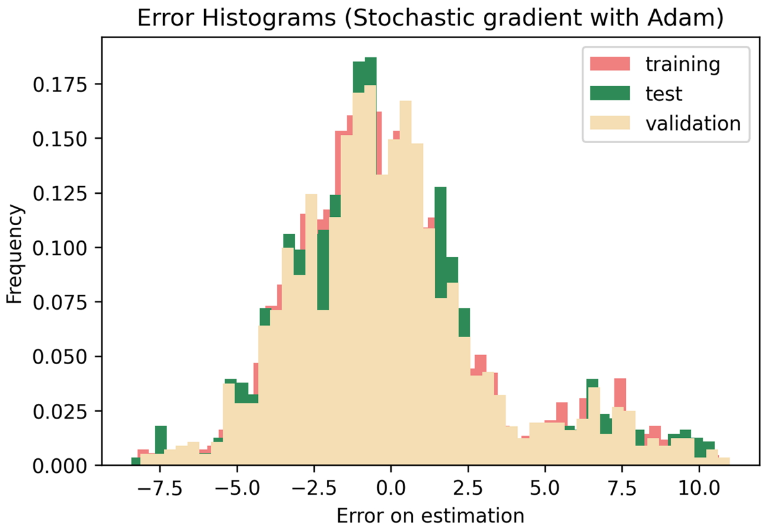
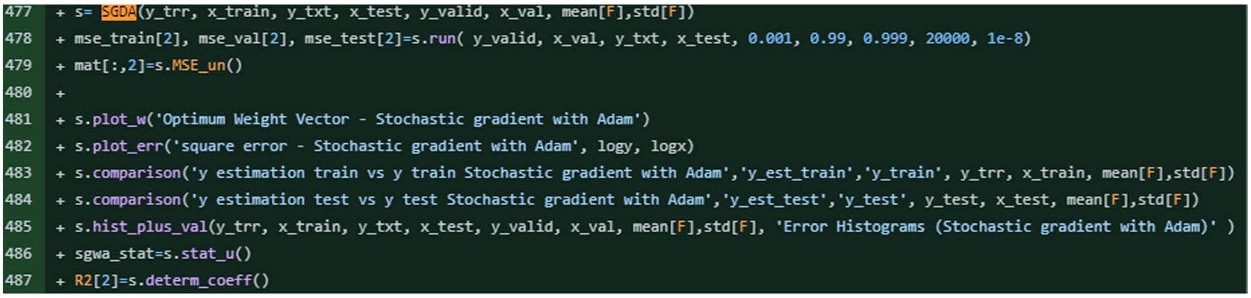


Fig. 16 Error histogram



```
+ s= SGDA(y_trr, x_train, y_txt, x_test, y_valid, x_val, mean[F],std[F])
+ mse_train[2], mse_val[2], mse_test[2]=s.run( y_valid, x_val, y_txt, x_test, 0.001, 0.99, 0.999, 20000, 1e-8)
+ mat[:,2]=s.MSE_un()
+
+ s.plot_w('Optimum Weight Vector - Stochastic gradient with Adam')
+ s.plot_err('square error - Stochastic gradient with Adam', logy, logx)
+ s.comparison('y estimation train vs y train Stochastic gradient with Adam','y_est_train','y_train', y_trr, x_train, mean[F],std[F])
+ s.comparison('y estimation test vs y test Stochastic gradient with Adam','y_est_test','y_test', y_test, x_test, mean[F],std[F])
+ s.hist_plus_val(y_trr, x_train, y_txt, x_test, y_valid, x_val, mean[F],std[F], 'Error Histograms (Stochastic gradient with Adam)' )
+ sgwa_stat=s.stat_u()
+ R2[2]=s.determ_coeff()
```

Fig. 17 Represents the part of the code, complete code is accessible at [29]

values. Ridge regression is utilized to reduce over-fitting concerns; when the noise is excessively huge, the optimum vector w may assume a tremendous value, leading to over-fitting. If $y = Xw + v$ has some large values of noise/error, the vector $\hat{w}$ may take huge values. Then, it might be convenient to solve the new problem, as stated in Eq. 31.

$$\min_{w} = ||y - Xw||^2 + \mu||w||^2 \tag{31}$$

μ should be set by trial and error. The gradient of the objective function is noted in Eq. 32.

$$\nabla_w g(w) = 2X^T Xw - 2X^T y_{meas} + 2\lambda w \tag{32}$$

By setting Eq. 32 equal to 0 leads to Eq. 33.

$$\hat{w} = (X^T X + \lambda I)^{-1} X^T y_{meas} \tag{33}$$

Figures 18, and 19 illustrate the Ridge Regression method's satisfactory performance. Motor UPDRS, like all other techniques except LLS, has the highest weight value, as stated in Fig. 20. Figure 21 illustrates the error histogram of the Ridge Regression, which follows the exact behavior of other methods.

Results and discussion

To analyze the methodologies and comprehend how it is feasible to anticipate the "regressand" and "regressor" in PD, MSE and R^2 were examined. Equation 34 provides the formulation of the MSE, which is commonly used to determine if the model's findings are satisfactory. A value closer to 0 consider a better assessment.

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2 \tag{34}$$

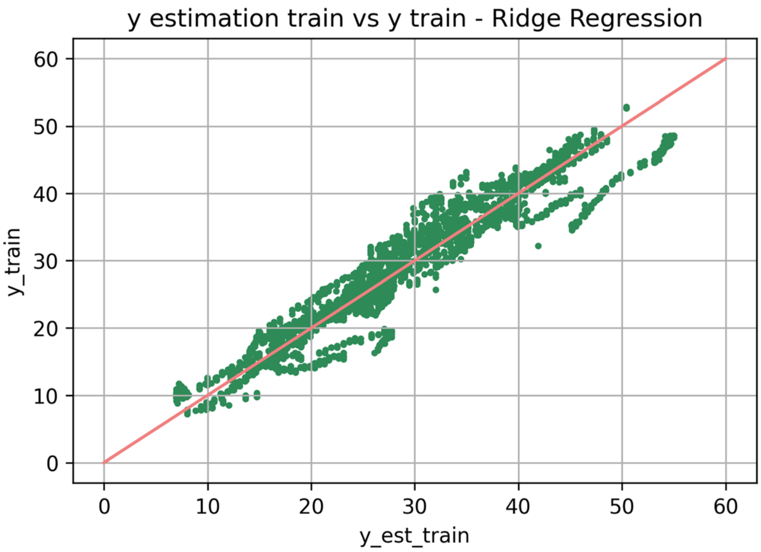


Fig. 18 y estimation train vs y train

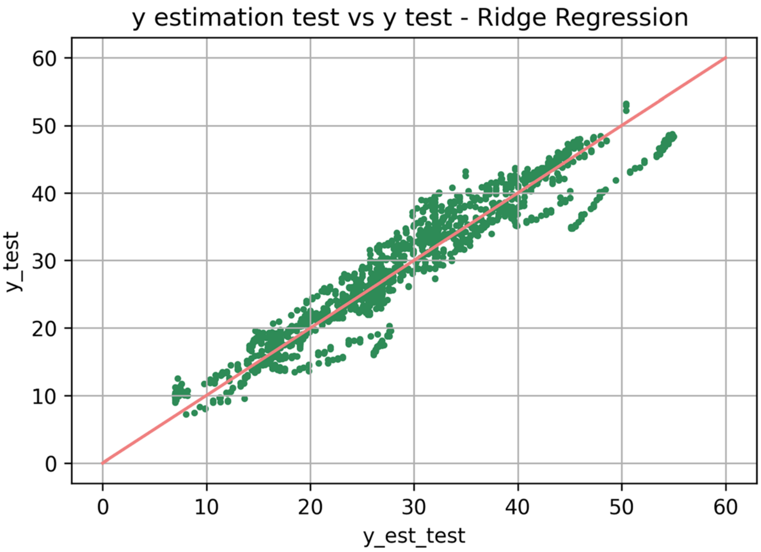


Fig. 19 y estimation test vs y test

On the other hand, also R^2 is considered, which is defined in Eq. 35.

$$R^2 = 1 - \frac{\sigma_e^2}{\sigma_y^2} = 1 - \frac{\sum_{n=1}^{N}[\hat{y}(n) - y(n)]^2}{\sum_{n=1}^{N}[y(n) - \bar{y}]^2} \tag{35}$$

Tables 3, 4, and 5 demonstrate that the solutions provide a more accurate assessment of the training set than the test set. In fact, the MSE of the training set is less than that of the test set. On the other hand, Table 6 represents the value of R^2, which should be as close as possible to 1.

In the Coefficient of determination, the value of 1.0 shows a perfect fit and a highly reliable model for the future, while a value of 0.0 would indicate the calculation fails to model the data accurately. From Table 6, the best value with Coefficient of determination is for the Adam optimization algorithm; however, it is very close to the result of the other methods. Behind acceptable results of methods and their ability to consider

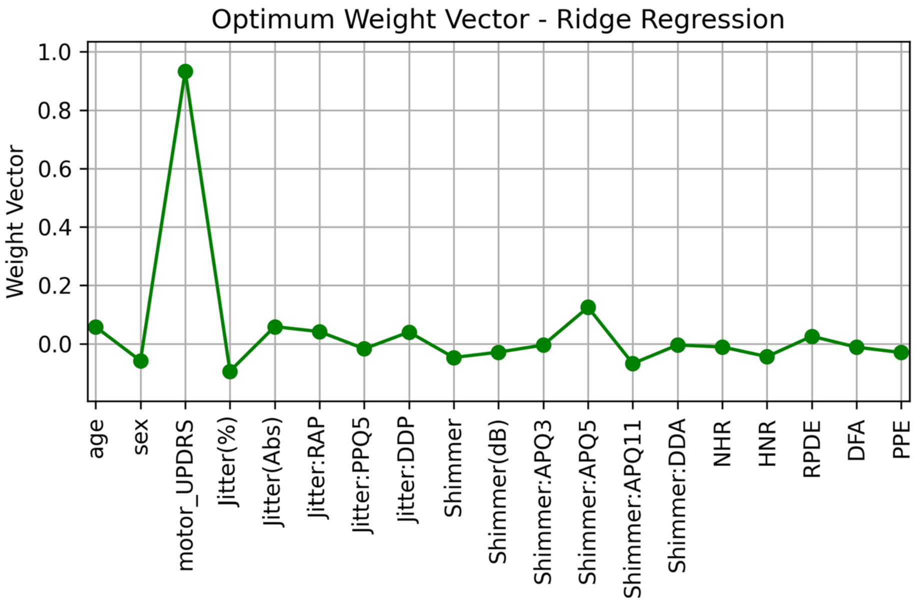


Fig. 20 Optimized weights for Ridge Algorithm

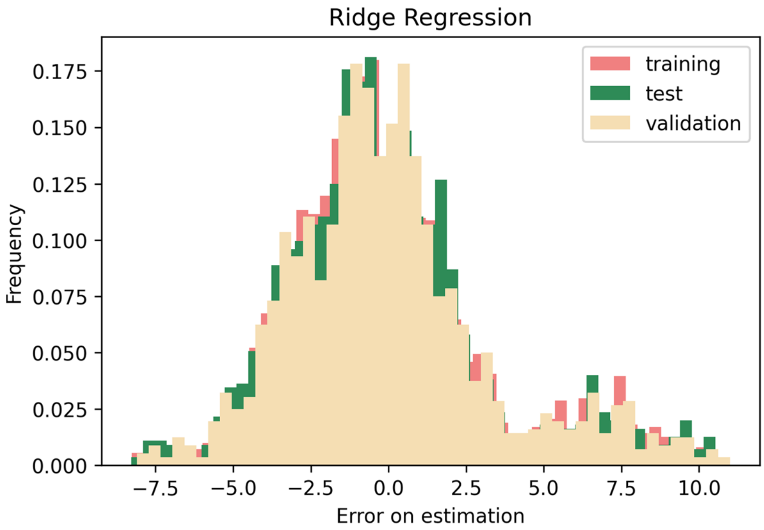


Fig. 21 Error histogram

Table 3 MSE results of training

Training	
Methods	**MSE**
LLS	10.4735
Conjugate Gradient	10.4762
Adam optimization algorithm	10.4762
Ridge Regressions	10.4798

Table 4 MSE results of the validation

Validation	
Methods	**MSE**
LLS	10.0138
Conjugate Gradient	10.5898
Adam optimization algorithm	10.5892
Ridge Regressions	10.5869

Table 5 MSE results of the test

Test	
Methods	**MSE**
LLS	10.9074
Conjugate Gradient	10.9075
Adam optimization algorithm	10.9072
Ridge Regressions	10.9136

Table 6 R^2 Score for different methods

Methods	R^2
LLS	0.904409
Conjugate Gradient	0.904408
Adam optimization algorithm	0.904410
Ridge Regressions	0.904354

them in the PD dataset, by assuming Cloud storage and Google Cloud, the algorithm's time complexity decreased dramatically and, on average, obtained 40 40 percent of the optimal time required to run methods. It means that not only is the development of techniques useful for researchers and doctors but also, using the concept of the Cloud enables programmers to develop algorithms more optimally.

Conclusion

By increasing the usage of ML, applying methods in various areas can solve issues and improve current assumptions. As in medical studies, because of the simplicity, some factors are considered steady parameters, and researchers and doctors have an easier understanding of the problem. Nevertheless, if more factors are assumed, they are interested because single-factor techniques have limited application. That is why the regression model is frequently used in such multi-factor situations. Linear regression models allow finding relationships between multiple factors to be defined and characterized. This research demonstrates that the most correlated feature to Total UPDRS is UPDRS Motor. This is predicted since PD manifests via the patient's movement, but the voice's characteristics negatively impact the Total UPDRS parameter's evaluation. Moreover, the ability of linear regression in the prediction of PD is another result obtained in this paper and confirms that by developing the Linear Regression methods, it is possible to predict PD, which is a disease without a cure, and prevention of it plays an important role. Ultimately, ML methods should be run and implemented in an environment that takes time to run optimally. That is why the concept of the Cloud is considered to reduce time complexity and enhance accuracy and accessibility. It is strongly advised that, for future expansion, various PD datasets be used and that a neural network be used in the cloud, followed by an evaluation of the methodologies. Moreover, in this regard, some recent and novel research on edge-computing task management has been indicated, which facilitates real-time processing [8] and [4].

Authors' contributions

Sahand Hamzehei: Idea, formulation, writing and programming. Omid Akbarzadeh: Formulation, motivation, writing and programming. Hani Attar: Experiments design. Khosro Rezaee: Literature investigation. Nazanin Fasihihour: Algorithms investigation. Mohammad R. Khosravi: Motivation and writing. All authors read and approved the final manuscript.

Funding

The authors declared that they had not received any financial support for this research.

Availability of data and materials

You can access this research code and data through the following link: https://github.com/SahandHamzehei/Regression-on-Parkinson-s-disease-data.

Declarations

Ethics approval and consent to participate

Not applicable.

Competing interests

The authors declare that they have no competing interests.

Received: 16 October 2022 Accepted: 28 December 2022
Published online: 21 January 2023

References

1. Keshavarz S, Keshavarz R, Abdipour A (2021) Compact active duplexer based on CSRR and interdigital loaded microstrip coupled lines for LTE application. Prog Electromagn Res C 109:27–37. https://doi.org/10.2528/PIERC20112307
2. Khosravi MR, Samadi S, Akbarzadeh O (2017) "Determining the optimal range of angle tracking radars," 2017 IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI), Chennai. pp. 3132-3135. https://doi.org/10.1109/ICPCSI.2017.8392303.
3. Gampala G, Reddy CJ (2020) Fast and Intelligent Antenna Design Optimization using Machine Learning. In: 2020 International Applied Computational Electromagnetics Society Symposium (ACES). pp. 1-2. https://doi.org/10.23919/ACES49320.2020.9196193
4. Chen Y, Zhao J, Wu Y, Huang J, Shen XS (2022) QoE-Aware Decentralized Task Offloading and Resource Allocation for End-Edge-Cloud Systems: A Game-Theoretical Approach. In: IEEE Transactions on Mobile Computing. https://doi.org/10.1109/TMC.2022.3223119
5. Keshavarz S, Kadry HM, Sounas DL (2021) Four-port Spatiotemporally Modulated Circulator with Low Modulation Frequency. In: 2021 IEEE Texas Symposium on Wireless and Microwave Circuits and Systems (WMCS). pp. 1-4. https://doi.org/10.1109/WMCS52222.2021.9493276
6. Huang J, Gao H, Wan S, Chen Y (2023) AoI-aware energy control and computation offloading for industrial IoT. Futur Gener Comput Syst 139:29-37. ISSN 0167-739X. https://doi.org/10.1016/j.future.2022.09.007
7. Akbarzadeh O (2022) Evaluating Latency in a 5G Infrastructure for Ultralow Latency Applications - Webthesis. http://webthesis.biblio.polito.it/id/eprint/22652. Accessed 14 Aug 2022

8. Chen Y, Gu W, Xu J (2022) Dynamic Task Offloading for Digital Twin-Empowered Mobile Edge Computing via Deep Reinforcement Learning. China Commun
9. Akbarzadeh O, Khosravi MR, Alex LT (2022). Design and Matlab Simulation of Persian License Plate Recognition Using Neural Network and Image Filtering for Intelligent Transportation Systems. ASP Trans Pattern Recognit Intell Syst 2(1):1-14. https://doi.org/10.52810/TPRIS.2021.100098
10. Leung MKK, Delong A, Alipanahi B, Frey BJ (2016) Machine Learning in Genomic Medicine: A Review of Computational Problems and Data Sets. In: Proceedings of the IEEE, vol. 104, no. 1, pp. 176-197. https://doi.org/10.1109/JPROC.2015.2494198
11. Li K, Zhao J, Hu J, Chen Y (2022) Dynamic energy efficient task offloading and resource allocation for NOMA-enabled IoT in smart buildings and environment. Build Environ 109513. ISSN 0360-1323. https://doi.org/10.1016/j.buildenv.2022.109513
12. Ma J (2020) Machine Learning in Predicting Diabetes in the Early Stage. In: 2020 2nd International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI). pp. 167-172. https://doi.org/10.1109/MLBDBI51377.2020.00037
13. Hamzehei S (2022) Gateways and Wearable Tools for Monitoring Patient Movements in a Hospital Environment - Webthesis. http://webthesis.biblio.polito.it/id/eprint/22711. Accessed 14 Aug 2022
14. Sruthi G, Ram CL, Sai MK, Singh BP, Majhotra N, Sharma N (2022) Cancer Prediction using Machine Learning. In: 2022 2nd International Conference on Innovative Practices in Technology and Management (ICIPTM). pp. 217-221. https://doi.org/10.1109/ICIPTM54933.2022.9754059
15. Akbarzadeh O, Baradaran M (2021) Khosravi MR (2021) IoT-Based Smart Management of Healthcare Services in Hospital Buildings during COVID-19 and Future Pandemics. Wirel Commun Mob Comput 2021(5533161):14. https://doi.org/10.1155/2021/5533161
16. Xu J, Li D, Gu W, Chen Y (2022) UAV-assisted task offloading for IoT in smart buildings and environment via deep reinforcement learning. Build Environ 222:109218. ISSN 0360-1323. https://doi.org/10.1016/j.buildenv.2022.109218
17. Akbarzadeh O, Khosravi MR, Shadloo-Jahromi M (2020) Combination of Pattern Classifiers Based on Naive Bayes and Fuzzy Integral Method for Biological Signal Applications. Curr Sig Transduct Ther 15(2) . https://doi.org/10.2174/1574362414666190320163953
18. Suzanne Brunt BM (2022) Parkinson's Disease: A Hopeful Future - Porterhouse Medical. Porterhouse Medical. www.porterhousemedical.com. https://www.porterhousemedical.com/news/parkinsons-disease-a-hopeful-future/#:~:text=Parkinsons%20disease%20is%20the%20fastest,an%20ageing%20population%20%5B1%5D
19. Marras C, Beck JC, Bower JH, Roberts E, Ritz B, Ross GW, Abbott RD, Savica R, Van Den Eeden SK, Willis AW, Tanner CM, on behalf of the Parkinson's Foundation P4 Group (2018) Prevalence of Parkinson's disease across North America. NPJ Park Dis 4(1):1–7. https://doi.org/10.1038/s41531-018-0058-0. https://www.nature.com/articles/s41531-018-0058-0
20. (2022) What Is Parkinson's? — Parkinson's Foundation. Parkinson's Foundation. https://www.parkinson.org/understanding-parkinsons/what-is-parkinsons . Accessed 20 Aug 2022
21. Tsanas A, Little MA, McSharry PE, Ramig LO (2010) Accurate telemonitoring of Parkinson's disease progression by noninvasive speech tests. IEEE Trans Biomed Eng 57(4):884–93. https://doi.org/10.1109/TBME.2009.2036000. Epub 2009 Nov 20 PMID: 19932995
22. Chen Y, Zhao F, Chen X, Wu Y (2022) Efficient Multi-Vehicle Task Offloading for Mobile Edge Computing in 6G Networks. In: IEEE Transactions on Vehicular Technology, vol. 71, no. 5, pp. 4584-4595. https://doi.org/10.1109/TVT.2021.3133586
23. (2022) Machine Learning and Cloud Computing - Javatpoint. (n.d.). https://www.javatpoint.com/machine-learning-and-cloud-computing. Accessed 20 Aug 2022.
24. Chen Y (2022) "Cost-Efficient Edge Caching for NOMA-Enabled IoT Services". CHINA COMMUNICATIONS/#/CHINA COMMUNICATIONS [[ZHONGGUO TONGXIN]]. https://libsearch.ncl.ac.uk/primo-explore/openurl?sid=google&=&=&auinit=Y&aulast=Chen&atitle=Cost-Efficient%20Edge%20Caching%20for%20NOMAEnabled%20IoT%20Services%E2%80%9D&title=China%20Communications&date=2022&issn=1673-5447&via_ncl=true&vid=NEWUI&institution=NCL&url_ctx_val=&url_ctx_fmt=null&isSerivcesPage=true
25. (2021) Google colab is a free cloud notebook environment - Biochemistry Computational Research Facility (BCRF) - UW-Madison. Biochemistry Computational Research Facility (BCRF). https://bcrf.biochem.wisc.edu/2021/02/05/google-colab-is-a-free-cloud-notebook-environment/#:~:text=Google%20Colab%20is%20a%20free%20cloud%2Dbased%20service%20that%20allows,and%20install%20new%20python%20libraries. Accessed 20 Aug 2022.
26. Qi L, Lin W, Zhang X, Dou W, Xu X, Chen J (2022) A Correlation Graph based Approach for Personalized and Compatible Web APIs Recommendation in Mobile APP Development. In: IEEE Transactions on Knowledge and Data Engineering. https://doi.org/10.1109/TKDE.2022.3168611
27. Burns, Ed (2021) What Is Machine Learning and Why Is It Important? SearchEnterpriseAI. https://www.techtarget.com/searchenterpriseai/definition/machine-learning-ML. Accessed 20 Aug 2022.
28. Zuehlke E (2015). Conjugate gradient methods - optimization. Conjugate Gradient Methods - Optimization. https://optimization.mccormick.northwestern.edu/index.php/Conjugate_gradient_methods. Accessed 20 Aug 2022.
29. Hamzehei S, Akbarzadeh O (2022) GitHub - SahandHamzehei/Regression-on-Parkinson-s-Disease-Data. GitHub. https://github.com/SahandHamzehei/Regression-on-Parkinson-s-disease-data. Accessed 3 Dec 2022

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.