

ALSFRS-R score prediction from sensor data using ML techniques

Ritesh Mehta^{1,*}, Aleksandar Pramov¹ and Shashank Verma¹

¹Georgia Institute of Technology, North Ave NW, Atlanta, GA 30332

Abstract

Amyotrophic Lateral Sclerosis (ALS) is characterized as a rapidly progressive neurodegenerative disease that presents individuals with limited treatment options in the realm of medical interventions and therapies. The disease showcases a diverse range of onset patterns and progression trajectories, emphasizing the critical importance of early detection of functional decline to enable tailored care strategies and timely therapeutic interventions. The present investigation, spearheaded by the iDPP@CLEF 2024 challenge, is centered around the utilization of sensor-derived data obtained through an app to construct a diverse array of machine learning models that are specifically designed to forecast the advancement of the ALS Functional Rating Scale-Revised (ALSFRS-R) score, leveraging the dataset provided by the organizers. In our analysis, multiple predictive models were evaluated to determine their efficacy in handling ALS sensor data. The temporal aspect of the sensor data was compressed and amalgamated using statistical methods, thereby augmenting the interpretability and applicability of the gathered information for predictive modeling objectives. The models that demonstrated optimal performance were a naive baseline and Elastic Net regression. The naive model achieved a Mean Absolute Error (MAE) of 0.20 and a Root Mean Square Error (RMSE) of 0.49, slightly outperforming the Elastic Net model, which recorded an MAE of 0.22 and an RMSE of 0.50. Our comparative analysis suggests that while the naive approach yielded marginally better predictive accuracy, the Elastic Net model provides a robust framework for understanding feature contributions.

Keywords

ALS, ALSFRS-R score prediction, sensor data analysis, Elastic Net regression, Predictive modeling in neurodegenerative diseases


1. Introduction


Amyotrophic Lateral Sclerosis (ALS), also known as Lou Gehrig's disease, is a progressive neurodegenerative disorder that affects motor neurons in the brain and spinal cord. The ALS Functional Rating Scale-Revised (ALSFRS-R), widely utilized in clinical and research settings, stands as a critical metric employed by healthcare professionals to evaluate the functional state of ALS patients. The precise forecasting of ALSFRS-R scores is essential for assessing disease progression and the effectiveness of therapeutic measures. Recent developments in sensor technology have opened up new avenues for continuous, non-invasive monitoring of ALS symptoms. The fusion of sensor data with predictive modeling presents the potential for more accurate and timely forecasts of disease progression, significantly benefiting patient care

CLEF 2024: Conference and Labs of the Evaluation Forum, September 9-12, 2024, Grenoble, France

*All authors contributed equally.

✉ rmehta307@gatech.edu (R. Mehta); apramov3@gatech.edu (A. Pramov); sverma342@gatech.edu (S. Verma)

 © 2024 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

and treatment management.

The iDPP@CLEF 2024 competition is an initiative aimed at leveraging sensor data and machine learning techniques to predict ALS progression. Task 1 involves predicting the ALSFRS-R scores assigned by medical professionals using sensor data collected via a dedicated app. Task 2 focuses on predicting self-assessment scores recorded frequently by patients. These tasks aim to enhance the accuracy and timeliness of ALS symptom monitoring and forecasting, providing valuable insights for patient care and treatment management.

To address the prediction of ALSFRS-R scores, we implemented several techniques. We started with a naive model to form a baseline and establish a reference for comparison. The naive model simply carries the last observed value forward. We then explored various Machine Learning algorithms for regression, as well as a Long Short-Term Memory (LSTM) neural network, to model the temporal dependencies in the sequential sensor data. Performance was evaluated via the Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) metrics identified by the challenge organisers.

The paper is structured as follows: Section 2 reviews related work, providing context and background on ALSFRS-R score prediction and the integration of sensor data with machine learning techniques. Section 3 details the methodology, including data processing steps and the modeling approaches we employed. Section 4 presents the results of our experiments, comparing the performance of different models. Section 5 discusses the implications of our findings. Section 6 outlines future work directions, and Section 7 concludes the study.

2. Related Work

In recent years, the integration of sensor data and machine learning (ML) techniques [1] has shown promising results in improving the accuracy and reliability of ALSFRS-R score predictions. [2] developed machine learning models to objectively measure ALS disease severity using voice samples and accelerometer data, while [3] further focuses specifically on deep learning methods to predict ALS disease progression. Outside of the ALS prediction field, [4] demonstrated the potential of temporal models in the healthcare domain by integrating EHR data; [5] applied various ML models to sensor data from accelerometers attached to dairy cattle for disease prediction and [6] demonstrates how temporal patterns from clinical and imaging data can be used to predict residual survival for cancer patients.

A more technical field of the literature deals specifically with the longitudinal aspect of the data and its affect on ML and DL methods. The advent of a wider application of ML methods in biomedical data in the recent years, brought the need to adapt traditional models for dealing with repeated measurements over time [7]. [8] extends Random Forests to handle fixed and random effects. [9, 10] give a more general overview of existing methods, while [11] focuses specifically on a neural network adaptation that can handle fixed and random effects.

3. Methodology

3.1. Data Description and Preprocessing

The raw features dataset provided for the competition consists of two parts: static data and sensor data. The former contains patient-specific baseline (constant) data, such as sex, age etc. The latter contains 91 time-series of patient-specific sensor data, collected over an average of nine months through a dedicated app developed by the BRAINTEASER project, using a fitness smartwatch in the context of clinical trials. For some patients, the clinical data starts before the app data, and some patients whose app data goes beyond the last observed clinical data. This required some synchronization between the clinical and the app data, in order to avoid look-ahead bias. Let t_1^s be the first sensor time point in the raw data, t_1^c be the first clinical time point in the raw data, as defined by the days from diagnosis for a given patient. Overall, we consider the time-overlap between clinical and sensor data. Some special cases are handled as follows:

- If $t_1^s > t_1^c$, then we discard clinical data beyond one step back from t_1^s . For example, consider a patient that has clinical observations at days 690, 780, 873, and sensor data from 800 onwards. We discard the clinical observation 690, but keep the rest. As we will see later, the reason for this is that we will use the previous clinical score as a feature and hence the first value to be predicted would be the target at day 873. That first value will be predicted using the sensor data between 800 and 872, as well as previous clinical visit from the 780th day since diagnosis.
- If $t_1^s \leq t_1^c$, then we use t_1^s as the starting point for the sensor data and we do not discard any clinical data.
- Sensor data after the last clinical observation is discarded.
- If the clinical data ends after the sensor data, we consider that point only if it is at maximum of 60 days from the last observed sensor data point. This is done so that we do not use too distant sensor data as a feature.
- Some patients have only one clinical observation. As we will use the previous clinical question responses in the set of features for the models, those patients will be discarded due to the missing previous response value.

Some patients had a few missing static data observations as well. Those were imputed by a simple median over all other patients for the respective feature.

The target variable (i.e., questionnaire responses) provides highly informative insights when visualized and analyzed over time, as shown in Figure 1. The visualization is done on the actual dataset we will use, i.e. after the pre-processing steps described above.

The curves reveal an initial period of slow degradation followed by a more rapid decline for many questions. We also notice that the scores change pretty slowly for most patients over the course of 100 days (the average number of days between consecutive visits to the clinician). Additionally, only a few patients manage to recover their scores to better levels. That observation has to be cautioned by the observation that the amount of patients with a longer follow-up decreases strongly over time. While at the beginning there are $n = 51$ observed patients (i.e. the full sample), this number drops to $n = 37$ for the second follow-up

and decreases further to just $n = 5$ and $n = 2$ for the fifth and sixth follow-up respectively. This is also evident by the higher CI bars on Figure 1.

A key insight here, as a result of above mentioned observations, is that the previous value of the score could be a useful engineered feature, which will also dictate the data pre-processing steps. Additionally, the heterogeneity in question types supports two modeling approaches: (1) treating the data as panel data with “question type” as a grouping variable for random effects, and (2) modeling each question separately.

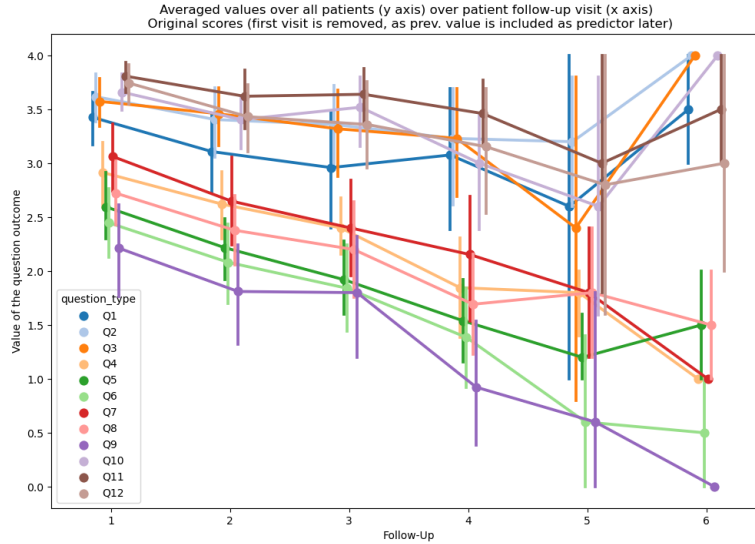


Figure 1: Averaged question response values for each follow-up time point. The x-axis shows the ordered visits by patient, whereas the y axis shows the average score. The vertical bars indicate a 95% confidence interval around the estimated mean. As we have fewer and fewer patients for longer follow-ups, the CI widen. There is some degree of variability with respect to the intercept among the questions, as well as the slopes. The beginning periods exhibit slow deterioration which already suggests that using the previous value as predictor for the next score will be useful. The points are jittered for visualization purposes.

3.2. Features generation

For our modeling, we considered three groups of features: *target-based*, *static* and *sensor-data*. While we used the static data *as-is*, we originally tried to build a baseline model based only on the target data and treat it as an autoregressive problem with some additional engineered simple transformations of the target dataset, e.g., the difference in days between two diagnosis, the first value of each patient, the previous value, one-hot encoding of the question type, the follow-up time-index (i.e. the x-axis in Figure 1). Based on this initial analysis, we quickly realized our initial hypothesis for the importance of the previous target value, stated in previous section, is valid, and we kept it as a feature for any subsequent modeling.

A central challenge in this study was the handling of the *sensor data* features. Those are observed daily (with some gaps), compared to the target (and related engineered variables such as the previous value), which is observed once every three months. To address this mismatch in the frequencies, we followed three different approaches:

Approach 1: By simply taking the median over a window of each sensor data column, where the window was defined to be between each two consecutive clinical visits - $[t - 1, t)$.

Approach 2: By generating a vast array of features derived from a window in each sensor data column, where the window is between each two consecutive clinical visits - $[t - 1, t)$, inspired by a feature-based time series analysis approach [12].

Approach 3: By using LSTM (Long Short Term Memory) cells for the sensor data and letting the network define the relevant transformations of the input.

Note that Approach 1 is a special and simple case of Approach 2. In both ways, we effectively express each sensor time window $[t - 1, t)$, with just one number. Doing this across all the windows of all patients builds the sensor data feature set. Approach 3 on the other hand does not have pre-defined (set of) transformation(s) on the input sensor data and we let the neural network itself pick up the relevant transformation of the input. Its details are provided in section 3.5 where the LSTM model is described.

For Approach 2, we employed the `tsfresh` python library for systematic feature engineering from time-series and other sequential data [13]. Simple examples of those can be e.g. the maxima/minima/median/mean etc. over each sensor data window per each patient between two clinical visits. A full outline of the feature extraction procedure and the considered extracted features can be found on the documentation website of the package. Note that, when one considers different parameter setups of the various extracted features, one would end up with multiple extracted features per time series. We have around 100 sensor time series and many of the extracted features by `tsfresh` had problems like high amount of missing values, low variability etc. and so were removed from the set of feature candidates. On each of those extracted features we performed filtering, by calculating `tsfresh` Spearmann correlation coefficient between the feature and the (one-step-ahead) clinical values per each question, over all patients. The hypothesis tests for significance are adjusted following the procedure in [14]. For the final feature set based on the app data, we experimented with either a) keeping all the features that were deemed significant per question or b) by fixing the number of the top k (e.g. 10) in terms of lowest p-value to keep in the sensor feature set. As the final modeling was done per each question separately, we settled on the former choice.

3.3. Modeling

3.3.1. CV setup

The small dataset posed several challenges with regards to choosing the appropriate cross-validation procedure for the hyperparameter tuning for our models. One key aspect is that the true (unseen) test set contains patients that are not present in the training set, rather than containing unseen data of the same patients. Hence, to assess the ability of our models to

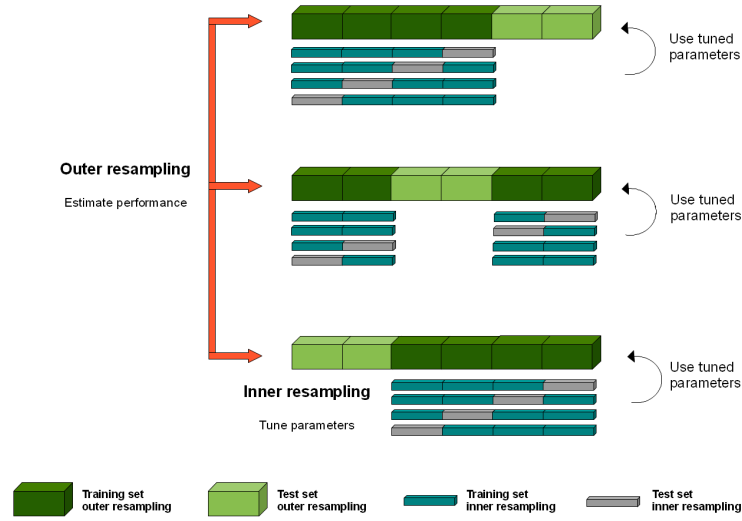


Figure 2: Cross-validation procedure for the hyperparameter tuning and final model fit in our work (Source: [17]). The original source illustrates the procedure without taking into account a grouped structure as in our dataset. Hence, in our work we adapted the procedure so that each patient’s complete data is contained in either the inner-train/test subsets, or the outer test-set.

generalize well, we used a nested k-fold cross validation strategy [15, 16]. This consists of two loops - an inner, and an outer one.

In each inner loop, we set aside a test set of 10% containing complete data of the patients (i.e. all observations). For the remaining data, we perform a further k-fold cross-validation, whereby we also adapted it to contain a complete set of observations of each patient in both the training and the validation sets. That ensured that there is no information leakage between different times of the patient’s data, as the true out of sample test (for the final submission) also contains data on unseen patients.

The outer loop then repeats the same procedure, but on another test set which covers another 10% of the patients. In total, we have 10 iterations in the outer loop, for each of which the RMSE is computed to allow for a model choice. The best hyperparameters overall are then chosen and all the data was fit using those prior the final submission.

Figure 4 illustrates the process for non-grouped data as described by [17]. The term “test inner-resampling” corresponds to what we call validation set. Overall, the additional “grouped” adaptation that we implemented to prevent information leakage, ensured that for any outer-loop/inner-loop combination, all the data for a patient is contained in either the (outer-loop) test, the (inner-loop) validation or the (inner-loop) train set.

3.3.2. Data augmentation

One of the major pain point in modeling for Task1 was the lack of sufficient datapoints for training as outlined previously. Given the high amount of features, this made the models more prone to overfitting . Fitting complex deep models would exacerbate this problem further. To

as well as deep learning models. We'll go over the details of LSTM modeling in the next section.

Naïve Model

As described in section 3.1.1, the scores for individual questions changes infrequently for an average patient over the course of two consecutive clinician visits (this is especially true since the scores are integers and don't allow partial progression from say 4 to 3). This suggests a Naive Model where the predicted score is the same as previous visit's score to serve as a baseline for all modeling approaches.

Modeling Approaches

In its given form, this is a multi-label multi-class classification problem. However, we can treat question_type as a covariate and transform this into a multi-class classification (with 12x training tasks). We can also train a separate model for each of the 12 questions which would also transform it into a multi-class classification problem.

Another way to formulate the problem is by noticing that in the training data 77% of the next visit's score is the same as that of the previous visit and of the remaining 23%, 14% has next visit's score = previous visit's score - 1. We can train an MLP classifier to learn the unchanging scores case from the rest. This becomes a simple binary classification problem (it requires post-processing to re-map the same vs diff prediction to actual scores).

The Overfitting problem

One thing that became clear in very early stages of modeling was that overfitting was going to be the most prevalent issue which was evident across a number of model classes like LinearRegression, RandomForest, kNNClassifier, LogisticRegression, GaussianNaiveBayes, GradientBoostingClassifier, etc. This points towards regularization being a strong requirement which is where models like ElasticNet and Lasso shine [18].

ElasticNet and Lasso are both modified LinearRegression models with regularization term in their objective function in addition to the loss term. ElasticNet minimizes the following expression:

$$1/(2 * n_samples) * ||y - Xw||_2^2 + l1_weight * ||w||_1 + l2_weight * ||w||_2^2$$

where $l1_weight = \alpha * l1_ratio$ and $l2_weight = 0.5 * \alpha * (1 - l1_ratio)$

and its special case is Lasso when $l1_ratio = 1.0$. An important parameter here is α , which is the regularization strength. In ElasticNet $l1_ratio$ allows application of penalty on L1 as well as L2 norm of w . We use GridSearch [19] to tune hyperparameters for ElasticNet and Lasso models with α and $l1_ratio$ being tunable hyperparameters for ElasticNet and just α for Lasso.

Training Data

The training data is formatted differently for different models, but for the above mentioned ElasticNet and Lasso models, the data was prepared to look as follows with all but last column as features and the last column as the target variable. Also note that this modeling was done

Patient ID	Days since Diagnosis	Previous Value	Delta days in Future	Sensor Feature1	...	Sensor featureN	Future Value
patient1	800	4	95	0.1	...	22	3
patient2	700	3	90	0.4	...	89	1
patient3	1400	2	120	0.2	...	43	2
patient4	300	4	110	0.9	...	09	4
patient5	500	4	100	0.0	...	51	2

Table 1
Representative training data after pre-processing.

on a per-question basis i.e. the *previous_value* and *future_value* in this table are for the question being modeled.

3.5. Modeling - LSTM

In our study, we employed a Long Short-Term Memory (LSTM) neural network to predict ALSFRS-R scores based on time-series sensor data. The LSTM model is well-suited for handling sequential data due to its ability to capture long-term dependencies. As depicted in Fig4, our model processes sensor data collected over multiple time points using a series of LSTM cells. Each cell captures temporal patterns in the data at different time steps, which are then fed into subsequent cells. To handle the uneven number of days between baseline and target scores, padding was applied to standardize the input sequences. The output from the final LSTM cell is concatenated with baseline scores and static patient data to enhance the model's predictive capability. This combined feature set is then passed through multiple linear layers, each dedicated to predicting one of the twelve ALSFRS-R sub-scores. This architecture allows the model to leverage both dynamic sensor inputs and static information, providing robust predictions for each functional domain assessed by the ALSFRS-R score.

4. Results

Details of how the data was prepared, processed, split, etc. has been explained in section 3. We used RMSE as the scoring function at all stages.

Task1

The results for Task1 are shown in Table2 below.

The table displays the RMSE of various models on each question on test data split from the train+val set. As can be seen here as well as Figure 5, *previous_value* has by far the most predictive power. For submitting, we ran Grid search using cross validation on entire training data (without the test split) and the model that gave the best validation set RMSE was selected

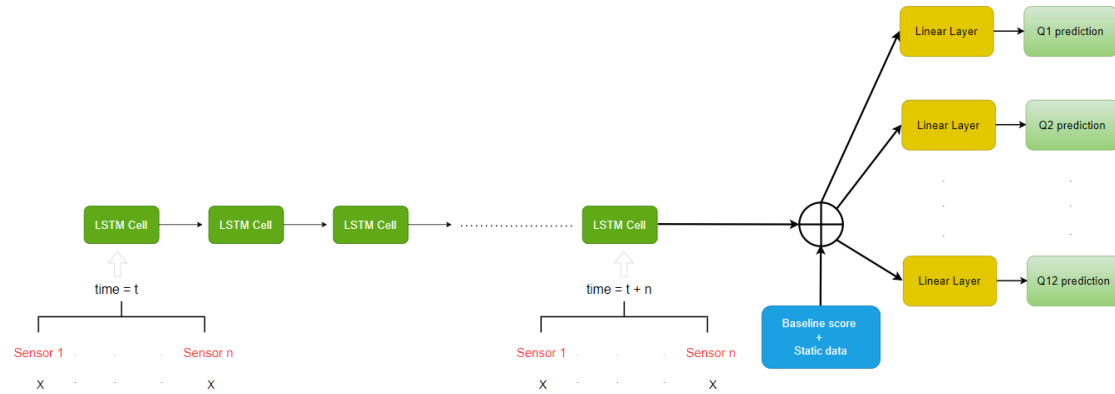


Figure 4: Architecture of the LSTM model used for predicting ALSFRS-R scores. The model processes time-series sensor data through sequential LSTM cells, capturing temporal dependencies. The output is combined with baseline scores and static patient data, then fed into multiple linear layers to predict each of the twelve ALSFRS-R sub-scores.

FS	Model	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12
-	Naive	0.47	0.38	0.43	0.65	0.69	0.77	0.74	0.62	0.78	0.5	0.46	0.54
W/o TsFresh	T1 EN	0.47	0.38	0.43	0.67	0.73	0.75	0.77	0.62	0.85	0.59	0.47	0.65
	T1 Lasso	0.47	0.38	0.43	0.66	0.73	0.79	0.79	0.64	0.79	0.55	0.49	0.54
	T1+2 EN	0.54	0.57	0.51	0.79	0.93	0.74	0.74	0.66	0.85	0.66	0.66	0.42
	T1+2 Lasso	0.54	0.56	0.49	0.66	0.85	0.7	0.69	0.64	0.86	0.71	0.66	0.42
W/ TsFresh	T1 EN	0.49	0.38	0.42	0.69	0.69	0.79	0.78	0.67	0.82	0.5	0.48	0.54
	T1 Lasso	0.47	0.38	0.42	0.71	0.69	0.8	0.77	0.67	0.81	0.52	0.46	0.46
	T1+2 EN	0.54	0.54	0.52	0.69	0.95	0.69	0.72	0.72	0.98	0.69	0.58	0.76
	T1+2 Lasso	0.57	0.55	0.52	0.67	0.8	0.69	0.73	0.69	0.88	0.7	0.64	0.72

Table 2

Task1's RMSE for various models separately modeling each of the 12 questions in ALSFRS-R scores. Green boxes denote the best performing model for the question. FS = FeatureSet, denoting whether median sensor features are used or TsFresh features are used (more details in section3.2. T1 denotes that only Task1 data was used for training whereas T1+2 denotes that Task1 data was augmented with that of Task2 for training (more details in section3.3.2.). EN and Lasso denote ElasticNet and Lasso models respectively.

for that question.

Using the above methodology, our final model was **ElasticNet + Naive** model. This achieved an RMSE of **0.5048** and MAE of **0.2222** on the final unseen test set and the leaderboard. This model was just shy of the Naive model only submission that we made which had an RMSE of **0.4912** and MAE of **0.2024**.

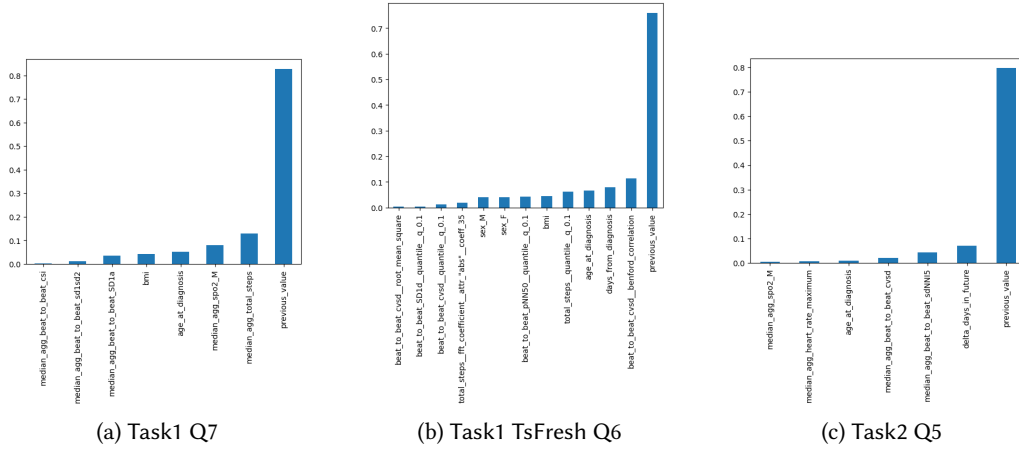


Figure 5: Feature Importance charts showing "previous_value" is consistently the most important feature.

Task2

Due to time constraints, we didn't manage to submit our model for Task2. However, using the ground truth that was released after the competition deadline, we ran our ElasticNet + Naive model and it gave an RMSE of **0.5594** and MAE of **0.2803**. This is better than the top leaderboard submission which is the Naive model with RMSE of 0.5774 and MAE of 0.2879.

5. Discussion

Discussion of the results and their implications.

6. Future Work

Future work will focus on clustering patient data to capture distinct patient phenotypes, allowing for more personalized predictions and treatment plans. We also plan to utilize freely available datasets such as PRO-ACT to enhance the robustness and generalizability of our models. Additionally, access to multimodal data sources such as audio data for speech, accelerometer data for muscle function, genetic data, imaging data, etc could help further improve the accuracy and reliability of ALSFRS-R score predictions. These efforts aim to refine our models and contribute to more effective ALS management and patient care.

7. Conclusions

Summary of the work and its contributions.

Acknowledgements

Thank you to the DS@GT CLEF team for their support.

References

- [1] A. S. Gupta, S. Patel, A. S. Premasiri, F. G. Vieira, At-home wearables and machine learning sensitively capture disease progression in amyotrophic lateral sclerosis, *Nature Communications* 14 (2023). URL: <https://api.semanticscholar.org/CorpusID:261062809>.
- [2] F. G. Vieira, S. Venugopalan, A. S. Premasiri, M. McNally, A. Jansen, K. McCloskey, M. P. Brenner, S. Perrin, A machine-learning based objective measure for als disease severity, *NPJ Digital Medicine* 5 (2022). URL: <https://api.semanticscholar.org/CorpusID:246240960>.
- [3] C. Pancotti, G. Birolo, C. Rollo, T. Sanavia, B. Di Camillo, U. Manera, A. Chiò, P. Fariselli, Deep learning methods to predict amyotrophic lateral sclerosis disease progression, *Scientific reports* 12 (2022) 13738.
- [4] E. Choi, M. T. Bahadori, A. Schuetz, W. F. Stewart, J. Sun, Doctor ai: Predicting clinical events via recurrent neural networks, *JMLR workshop and conference proceedings* 56 (2015) 301–318. URL: <https://api.semanticscholar.org/CorpusID:5842463>.
- [5] G. Vidal, J. Sharpnack, P. Pinedo, I. C. Tsai, A. R. Lee, B. Martínez-López, Comparative performance analysis of three machine learning algorithms applied to sensor data registered by a leg-attached accelerometer to predict metritis events in dairy cattle, volume 4, *Frontiers*, 2023, p. 1157090.
- [6] N. F. Smedley, B. M. Ellingson, T. F. Cloughesy, W. Hsu, Longitudinal patterns in clinical and imaging measurements predict residual survival in glioblastoma patients, *Scientific Reports* 8 (2018). URL: <https://api.semanticscholar.org/CorpusID:52846175>.
- [7] J. Pinheiro, D. Bates, *Mixed-effects models in S and S-PLUS*, Springer science & business media, 2006.
- [8] A. Hajjem, F. Bellavance, D. Larocque, Mixed-effects random forest for clustered data, *Journal of Statistical Computation and Simulation* 84 (2014) 1313–1328.
- [9] A. Cascarano, J. Mur-Petit, J. Hernandez-Gonzalez, M. Camacho, N. de Toro Eadie, P. Gkontra, M. Chadeau-Hyam, J. Vitria, K. Lekadir, Machine and deep learning for longitudinal biomedical data: a review of methods and applications, *Artificial Intelligence Review* 56 (2023) 1711–1771.
- [10] S. Hu, Y.-G. Wang, C. Drovandi, T. Cao, Predictions of machine learning with mixed-effects in analyzing longitudinal data under model misspecification, *Statistical Methods & Applications* 32 (2023) 681–711.
- [11] T. Wörtwein, N. B. Allen, L. B. Sheeber, R. P. Auerbach, J. F. Cohn, L.-P. Morency, Neural mixed effects for nonlinear personalized predictions, in: *Proceedings of the 25th International Conference on Multimodal Interaction*, 2023, pp. 445–454.
- [12] B. D. Fulcher, *Feature-based time-series analysis*, in: *Feature engineering for machine learning and data analytics*, CRC press, 2018, pp. 87–116.
- [13] M. Christ, N. Braun, J. Neuffer, A. W. Kempa-Liehr, Time series feature extraction on

basis of scalable hypothesis tests (tsfresh—a python package), *Neurocomputing* 307 (2018) 72–77.

- [14] M. Christ, A. W. Kempa-Liehr, M. Feindt, Distributed and parallel time series feature extraction for industrial big data applications, *CoRR* abs/1610.07717 (2016). URL: <http://arxiv.org/abs/1610.07717>. arXiv:1610.07717.
- [15] G. C. Cawley, N. L. Talbot, On over-fitting in model selection and subsequent selection bias in performance evaluation, *The Journal of Machine Learning Research* 11 (2010) 2079–2107.
- [16] M. Kuhn, K. Johnson, Feature engineering and selection: A practical approach for predictive models, Chapman and Hall/CRC, 2019.
- [17] V. Lyashenko, A. Jha, Cross-validation in machine learning: How to do it right, 2024. URL: <https://neptune.ai/blog/cross-validation-in-machine-learning-how-to-do-it-right>, accessed: 2024-05-24.
- [18] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning, Springer Series in Statistics, Springer New York Inc., New York, NY, USA, 2001.
- [19] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine learning in Python, *Journal of Machine Learning Research* 12 (2011) 2825–2830.