Performing Supervised Learning on Monkeytype Logs

Ashutosh Rai
2422308
Data Science Lab

Contents

1 Introduction

The ability to quantify and predict human skill acquisition is a valuable task. This project focuses on data from Monkeytype, a popular typing test website, to determine if a user's future performance can be modeled based on their past sessions. The dataset consists of a single user's performance metrics over 644 sessions, recorded chronologically.

The initial goal was to solve a **regression problem**: to predict the exact Words Per Minute (WPM) a user would achieve in their next session. This report details the extensive experimental process, beginning with this regression task, and explains the data-driven pivot to a **classification problem** after initial models failed to find a strong predictive signal.

This report is structured as follows:

- **Methodology:** Details the common data preparation, feature engineering, and evaluation strategy used.
- **Regression Approach:** Summarizes the results from all regression models and establishes their inability to solve the problem.
- **Pivot to Classification:** Explains the reframing of the problem and details the comparative results of multiple classification models.
- **Results** Provides a deep dive into the best-performing model, its performance metrics, and feature importances.
- **Conclusion:** Discusses the final findings and the reasons for the model's performance limitations.

2 Methodology

2.1 Data Preparation and Feature Engineering

All experiments began with the same pre-processed dataset. The first critical step was to sort the data chronologically by its `datetime` stamp. This is essential for any time-series task to prevent data leakage.

Based on the raw data (WPM, accuracy, consistency), a set of features was engineered to capture the user's recent performance trends. While a large set of 44 features was initially tested, analysis of feature importances led to the use of a hyper-focused set of 6-7 core features for most experiments:

- **`wpm`, `acc`, `consistency`:** The raw metrics from the current session.
- **`wpm_ewm3`:** An exponentially-weighted moving average (span=3).
- **`wpm_roll3_mean`:** The rolling average WPM over the last 3 sessions.
- **`wpm_cummean`:** The cumulative mean WPM over the user's entire history.

Additional features, such as cyclical `time_of_day` and `time_since_last_session`, were also tested.

2.2 Evaluation Strategy

A 90/10 time-based split was used, allocating the first 90% of the data for training and the final 10% for testing. This ensures the model is always predicting the "future" it has not seen.

For hyperparameter tuning, standard k-fold cross-validation is inappropriate as it shuffles data, breaking the temporal order. Therefore, all `GridSearchCV` processes were conducted using `TimeSeriesSplit`. This method creates folds that respect the data's chronological sequence (for example, train on [1, 2, 3], validate on [4]; train on [1, 2, 3, 4], validate on [5]), providing a robust and realistic measure of a model's generalization performance.

3 Regression Approach

The initial objective was to predict the exact `y_next` (the WPM of the next session). Two simple baselines were established for comparison.

3.1 Regression Baselines

1. **Previous WPM:** Assumes the next WPM will be identical to the current WPM.
2. **Rolling-3 WPM:** Assumes the next WPM will be the average of the last 3 sessions.

3.2 Models and Results

Four models were trained on the focused 6-feature set to predict the log-transformed WPM (`np.log1p(y_next)`), which helps stabilize variance. The test set performance is summarized in Table 1.

Table 1: Performance of Regression Models on Test Set

Model	MAE	RMSE	Accuracy (±3.0 WPM)
Baseline (Previous WPM)	6.730	8.905	-
Baseline (Rolling-3 WPM)	5.534	7.167	-
Linear Regression	5.078	6.655	43.08%
Random Forest Regressor	5.210	6.840	41.54%
LightGBM Regressor	5.053	6.681	46.15%
XGBoost Regressor	5.649	7.431	43.08%

3.3 Regression Analysis

The results were conclusive: **none of the complex, tuned models could significantly beat the simple Rolling-3 WPM baseline.** The Linear Regression model performed best, but its 43% accuracy was still very low. Furthermore, the tree-based models (LGBM and XGBoost) produced logs filled with `No further splits with positive gain` warnings, indicating they could not find a clear signal.

The feature importance plots (Figures 1-3) all showed that models were overwhelmingly reliant on the most recent WPM, with other features providing little value. This, combined with the poor performance, strongly suggested that predicting the **exact WPM** is not feasible with this data, as the target variable is too noisy and influenced by uncaptured external factors.

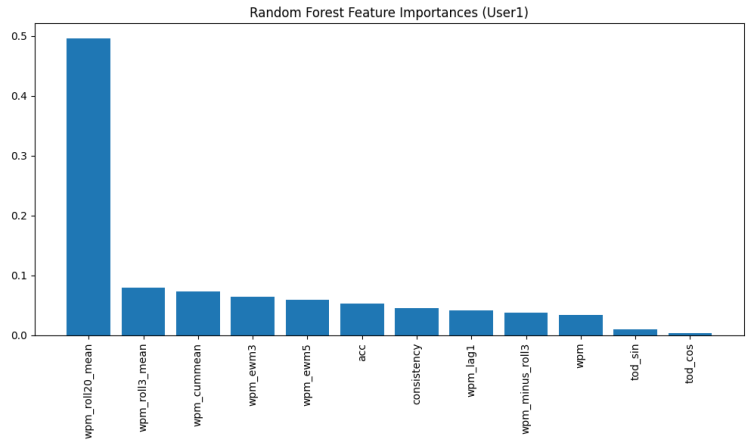

Figure 1: Feature importances for the Random Forest Regressor.

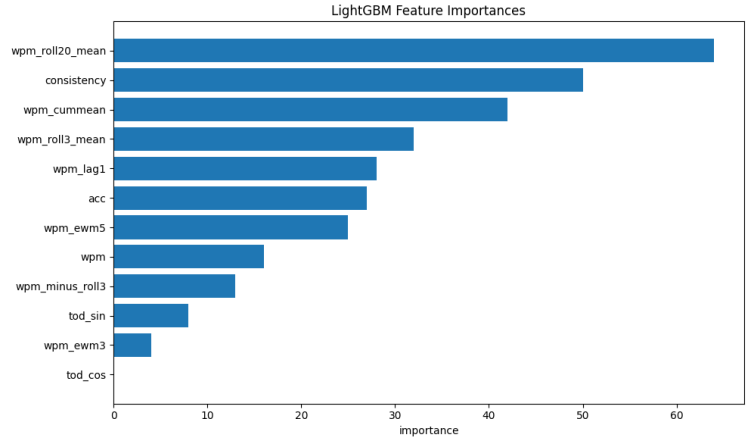

Figure 2: Feature importances for the LightGBM Regressor.

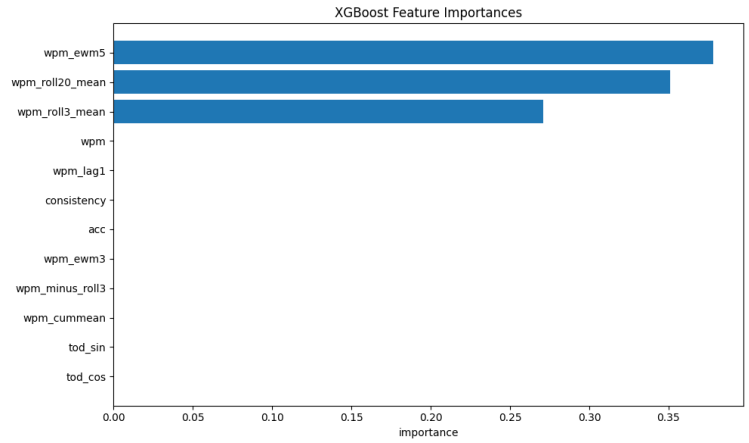

Figure 3: Feature importances for the XGBoost Regressor.

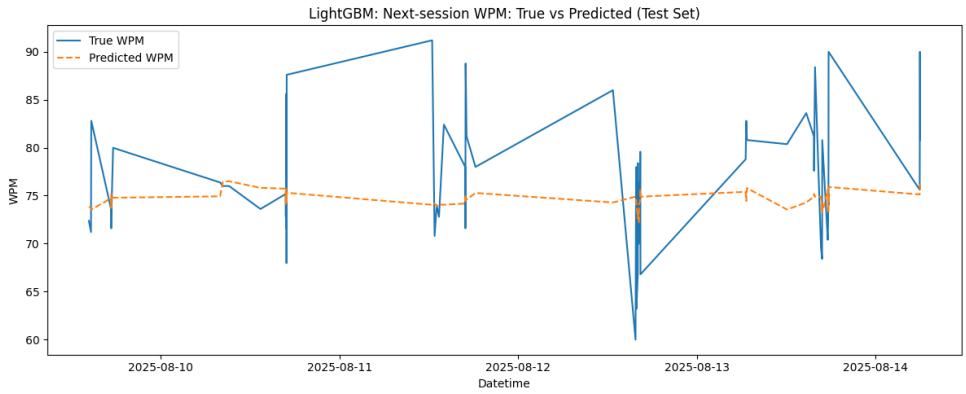

Figure 4: LGBM Regressor predictions vs. true WPM values on the test set.

4 Pivot to Classification

Based on the failure of the regression approach, the problem was reframed. Instead of predicting a precise value, a simpler, more robust question was asked: **Will the user's performance Increase, Decrease, or stay Stable?**

A new target variable was created by binning the change in WPM (`wpm_delta`):

- **Increase:** `wpm_delta > 3.0`
- **Decrease:** `wpm_delta < -3.0`
- **Stable:** `wpm_delta` is between -3.0 and 3.0

A check of the training data revealed a balanced dataset (Increase: 204, Decrease: 196, Stable: 179), confirming that any modeling difficulty would be due to signal weakness, not class imbalance. This finding meant that oversampling techniques like SMOTE were unnecessary.

4.1 Model Comparison

Multiple models were tested on this new classification task. All were tuned using `GridSearchCV` with `TimeSeriesSplit`. The performance on the held-out test set is summarized in Table 2.

Table 2: Performance of Classification Models on Test Set

Model / Experiment	Test Accuracy
LightGBM Classifier (6-feature)	**69.23%**
LGBM Classifier (7-feature)	61.54%
Logistic Regression (Poly)	60.00%
Support Vector Classifier	55.38%

The **LightGBM Classifier with the 6-feature set** was the clear winner, achieving the highest accuracy. Further attempts to improve this score by adding new features (like `time_since_last_session`) or using different models (SVC, Logistic Regression) resulted in lower performance, confirming that this 6-feature model was the most optimal.

5 Results

The final, best-performing model was a **LightGBM Classifier** trained on the 6-feature set with robust `TimeSeriesSplit` cross-validation. The final model's best-performing parameters, found via `GridSearchCV`, resulted in a simple and constrained model, appropriate for this dataset's weak signal.

5.1 Performance Metrics

The model's final performance on the test set (Table 3) provides a nuanced picture. The overall accuracy of **69.23%** is a significant lift from a 38.5% baseline (predicting the majority class, 'Increase').

Table 3: Classification Report for Final LGBM Model

Class	Precision	Recall	F1-Score	Support
Decrease	0.74	0.89	0.81	19
Increase	0.68	0.60	0.64	25
Stable	0.65	0.62	0.63	21
Weighted Avg	**0.69**	**0.69**	**0.69**	**65**

The model's strongest capability is identifying when a user is about to perform worse. It correctly identified **89% of all "Decrease" sessions** (Recall = 0.89), making it a reliable indicator of negative trends.

5.2 Confusion Matrix Analysis

The confusion matrix in Table 4 details the model's specific successes and failures.

Table 4: Confusion Matrix for Final LGBM Model

	Predicted: Decrease	Predicted: Stable	Predicted: Increase
Actual: Decrease	**17**	1	1
Actual: Stable	2	**13**	6
Actual: Increase	4	6	**15**

The matrix confirms the findings that the model is highly effective on the "Decrease" class (17 correct, only 2 misclassified). Its main area of confusion lies in distinguishing between "Stable" and "Increase" sessions, which it mistakes for each other.

5.3 Feature Importance

The feature importance plot (Figure 5) for the final model confirms that recent performance metrics are the most predictive, with `wpm` (current WPM), `wpm_ewm3` (short-term trend), and `consistency` being the top 3 drivers.

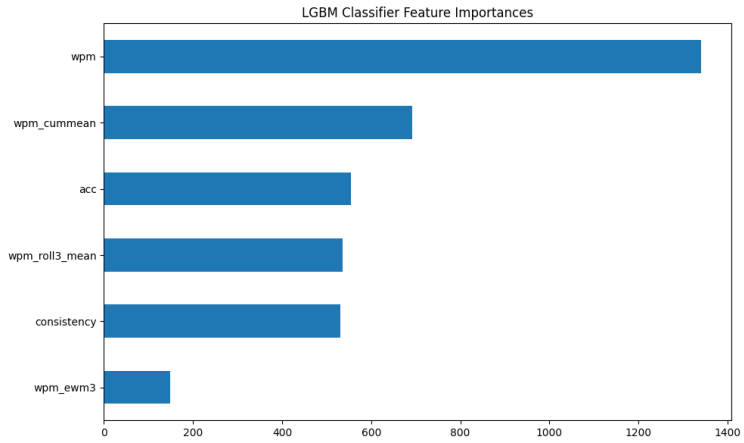

Figure 5: Feature importances for the final LGBM Classification model.

6 Conclusion

The primary challenge of this project was not model selection, but problem definition. The initial regression task failed because the target (exact WPM) is too noisy, with performance being highly variable from one session to the next. This variability is likely driven by human factors (fatigue, focus, mood) that are not captured in the dataset.

By pivoting to a classification problem, we successfully built a model that can predict the **direction** of performance with **69.23% accuracy**. This is a strong, positive result that significantly outperforms a naive baseline.

The model's limitations, such as the confusion between "Stable" and "Increase", are not a failure of the model but a reflection of the data itself. The signal is weak, and our iterative experiments show we have likely reached the performance ceiling for this specific feature set and dataset size.

Finally, simply augmenting the data (for example, with SMOTE) was shown to be ineffective. This is because the data is a time-series, and its chronological structure is paramount. Such techniques would only add noise and destroy the very temporal patterns we are trying to model.