Detecting working patterns from online workers and predicting task completion

Janani Chitra Rajendran jrajend@clemson.edu
Pritham Vinay Jujjavarapu pjujjav@g.clemson.edu
Sahithi Tatineni sahitht@g.clemson.edu

Abstract

As remote work continues to expand across digital labor platforms such as MTurk, Fiverr, and Upwork, understanding online worker behavior has become critical for enhancing productivity, engagement, and operational efficiency. This project aims to predict task completion times by analyzing user interaction data including browser events, platform usage, task subtypes, and temporal activity. Using a dataset comprising approximately 3.5 million observations, extensive preprocessing and feature engineering were performed to convert raw behavioral logs into meaningful predictors. A Random Forest regression model was initially developed as a robust baseline, followed by the implementation of an XGBoost model to improve predictive accuracy. Through hyperparameter optimization and model evaluation using RMSE and MAE metrics, XGBoost significantly outperformed the baseline, achieving near-zero prediction error. The results demonstrate the potential of machine learning in modeling online work patterns and provide actionable insights for stakeholders aiming to optimize workforce management and task scheduling in gig economy platforms.

1 Introduction

In the evolving landscape of remote and online work, platforms like MTurk, Fiverr, Freelancer, and Upwork host a growing population of digital laborers engaging in diverse microtasks. However, stakeholders including platform administrators, researchers, and employers face a crucial challenge: understanding and predicting task completion times based on user activity patterns. Such insight is vital for optimizing task allocation, improving worker experience, and enhancing platform efficiency.

The core problem addressed by this project is the lack of predictive intelligence surrounding how long online tasks take to complete, based on behavioral data such as browser events, platform activity, and task metadata. The absence of accurate task time estimation impedes platform optimization, hinders efficient scheduling, and may result in suboptimal resource management. Stakeholders are increasingly recognizing the need for data-driven modeling techniques to predict such outcomes and inform interventions.

Solving this challenge has broad implications. Accurate prediction of task completion times allows platforms to personalize task recommendations, identify disengaged users early, and optimize task design. For workers, it can reduce cognitive overload and improve earnings through better task selection. For researchers, the ability to model engagement patterns provides a window into user behavior dynamics within gig economies.

The objective of this project is to predict the time of day a task is likely to be completed by modeling user interaction patterns. Specifically, the output variable is the hour during which a task-related event occurs, inferred through engineered features from the dataset. To achieve this, two machine learning models—Random Forest Regression and XGBoost Regression were employed. The analysis focuses on identifying significant behavioral predictors and optimizing model performance using advanced preprocessing, encoding techniques, and hyperparameter tuning.

The rationale for this approach is grounded in the hypothesis that platform usage behaviors and task metadata encode temporal patterns of user engagement. By accurately modeling these patterns, the project seeks to equip stakeholders with tools to make evidence-based decisions. For instance, understanding when workers are most active and productive can inform scheduling algorithms or worker engagement strategies.

This approach aligns with prior work such as Fatima et al. (2023) [1], which highlights the superior performance of XGBoost in modeling non-linear behavioral data, validating its selection in this project for improving over the Random Forest baseline. The layered methodological progression from data cleaning and exploratory analysis (Checkpoint 1), to Random Forest modeling (Checkpoint 2), and finally to XGBoost optimization (Checkpoint 3) ensures a thorough and scientifically rigorous exploration of the problem.

2 Methods

2.1 Data Source and Description

The dataset used for this study was provided as part of the course curriculum and represents behavioral interaction logs collected from various online crowd work platforms. These platforms include Amazon Mechanical Turk (MTurk), Fiverr, Freelancer, Upwork, and other unnamed services commonly used for digital task outsourcing. Although the precise data collection mechanism was not disclosed, the dataset appears to have been derived from browser-based activity tracking systems that captured real-time interactions performed by users while completing microtasks on these platforms.

The data spans a fifteen-day period from May 9 to May 23, 2020, and comprises a total of 3,496,373 records. Each observation corresponds to a discrete user action, such as loading a webpage, starting a task, focusing or blurring the browser, and other relevant interactions. These events are enriched with metadata including platform type, task subtype, browser event type, and user identifier. The dataset includes 255,518 unique URLs, 18 browser event types, 29 task subtypes, and 10 higher-level task categories. In total, activity from 119 unique users is captured across all platforms.

The dataset was initially unstructured, with missing column headers and inconsistent formatting. Preprocessing efforts revealed that the user column had 264 missing entries. To preserve the completeness of the dataset, missing values in categorical fields were imputed with a placeholder label, “Unknown.” Numerical missing values were filled using median imputation to avoid skewing the distribution and to maintain central tendency. Time information was originally recorded in milliseconds and was later converted into a human-readable datetime format for temporal analysis. Further transformation was performed to derive cyclic time features such as hour of day, which was encoded using sine and cosine functions to capture periodicity.

The final dataset used for modeling was structured in tabular format, containing both categorical and numerical variables. It was free of null entries after imputation and ready for downstream machine learning tasks. This robust and diverse dataset enabled the extraction of meaningful behavioral patterns and facilitated the development of accurate predictive models for task completion time estimation.

2.2 Data Preprocessing

To ensure the dataset was suitable for predictive modeling, a series of structured and methodical preprocessing steps were undertaken. The initial dataset lacked column headers, which were assigned based on the accompanying data dictionary to facilitate consistent referencing and manipulation. The time column, which recorded timestamps in milliseconds, was transformed into a standard datetime format using the pandas.to_datetime function. This conversion enabled the extraction of time-based features, particularly the hour of task activity, which became the target variable for subsequent modeling.

Missing data was addressed with tailored imputation strategies. In categorical columns such as platform, task-subtype, and browser-event, missing values were replaced with the placeholder label “Unknown,” ensuring that the data's structural integrity was preserved without introducing bias. For numerical attributes, the median value of each respective column was used to replace missing entries. This approach was chosen over mean imputation to mitigate the influence of potential outliers and maintain the central tendency of the data distribution.

Following imputation, categorical variables were transformed using one-hot encoding. This process generated binary indicator variables for each category within features such as browser-event, platform, task-subtype, type, and user. One-hot encoding allowed the models to interpret these non-numeric values without assuming any ordinal relationship between categories. The numerical features were then standardized using the StandardScaler to ensure that all variables had equal influence during model training, thereby avoiding the risk of features with larger numerical ranges overpowering those with smaller scales.

An important feature transformation involved capturing the cyclical nature of time. Since the hour feature exhibits periodicity over a 24-hour cycle, it was encoded using sine and cosine transformations. Specifically, the hour value was normalized to a $[0, 2\pi]$ scale and transformed into two new features: hour_sin and hour_cos allowing machine learning models to better capture temporal dependencies without misinterpreting adjacent hours as being numerically distant (e.g., 23 and 0).

Once the preprocessing and feature transformation were completed, the dataset is divided into, assigning 80% for training purposes and 20% for testing. This split ensured a consistent and unbiased evaluation of model performance. The comprehensive preprocessing pipeline enabled the extraction of meaningful signals from raw user interaction logs and ensured compatibility with tree-based machine learning models employed in this project.

2.3 Baseline Analysis/Model

The baseline model selected for this project was a Random Forest Regressor, a widely-used ensemble learning method known for its robustness and capacity to handle both categorical and numerical data. This model was chosen as the initial predictive framework to estimate the hour at which a task-related event occurs, based on user behavior and platform metadata. Random Forest works by building an ensemble of decision trees during the training phase and combining their outputs to produce a final prediction. This method reduces variance and mitigates overfitting, making it especially effective for high-dimensional datasets with intricate feature relationships.

The rationale for selecting Random Forest as the baseline method lies in its strong performance across a variety of predictive tasks and its ability to model non-linear relationships without requiring extensive data transformations. Given that the dataset contained millions of records with a mix of structured and semi-structured variables such as browser events, task types, and platform usage the Random Forest algorithm offered a straightforward yet effective solution. Moreover, its ability to provide feature importance rankings added value to the exploratory phase, enabling the identification of the most influential predictors in user behavior.

To enhance model performance and reliability, a grid search procedure with three-fold cross-validation was employed to tune key hyperparameters. The hyperparameter search space included variations in the number of estimators, the depth of each tree, and node-splitting criteria. Specifically, the number of estimators was tested at 100, 200, and 300 trees. The maximum depth parameter was varied among 10, 20, 30, and unrestricted settings. Minimum sample thresholds for both node splitting (min_samples_split) and leaf creation (min_samples_leaf) were also tuned, with tested values of 2, 5, and 10 for the former and 1, 2, and 4 for the latter.

Following this systematic tuning process, the optimal configuration for the baseline Random Forest model was determined to include 300 estimators, the model was configured with a maximum tree depth of 10, required at least 10 samples to split a node, and ensured a minimum of 4 samples in each leaf node. This configuration achieved a strong balance between bias and variance, offering a reliable foundation for performance comparison against more advanced modeling techniques in the subsequent phases of the project.

2.4 Alternative Analysis/Model

To address the limitations observed in the baseline Random Forest model and improve the predictive performance, an alternative model using the XGBoost Regressor was implemented. XGBoost, short for Extreme Gradient Boosting, is an advanced tree-based ensemble method that sequentially builds decision trees in a gradient boosting framework, with each tree attempting to correct the errors of its predecessor. Unlike Random Forest, which operates in parallel and relies on averaging, XGBoost emphasizes model refinement through weighted boosting, enabling it to capture complex and subtle patterns in high-dimensional data more effectively.

XGBoost was chosen as the alternative approach due to its strong track record in structured data challenges, its effectiveness in capturing non-linear patterns, and its integrated regularization features that help improve model generalization, which reduce the risk of overfitting. Additionally, XGBoost is computationally optimized and supports native handling of missing values, making it highly suitable for the behavioral dataset used in this project. Given that the dataset consists of millions of interactions and a large number of categorical and engineered features, XGBoost offered a scalable and accurate solution to further reduce prediction error.

To fully leverage the potential of XGBoost, an extensive hyperparameter optimization process was conducted using RandomizedSearchCV, which allowed for efficient exploration of a broad parameter space. Several key hyperparameters were considered during tuning. The learning rate, which controls the step size at each boosting iteration, was tested with values of 0.01, 0.05, 0.1, and 0.2. The maximum depth of trees was evaluated at values of 3, 5, 7, and 10 to control the model complexity. The number of estimators, representing the number of boosting rounds, was tested at 100, 200, and 300. In addition, the subsample ratio and column sampling ratio (colsample_bytree) were varied between 0.6, 0.8, and 1.0 to introduce randomness and prevent overfitting. XGBoost's regularization parameters, reg_alpha (L1 penalty) and reg_lambda (L2 penalty), were tuned using values of 0, 0.1, and 1 for reg_alpha, and 1, 2, and 3 for reg_lambda.

The final model configuration that emerged from this randomized search includes a learning rate of 0.1, a maximum depth of 7, 200 estimators, a subsample ratio of 0.8, and regularization terms of reg_alpha = 0.1 and reg_lambda = 2. This combination offered a well-calibrated balance between learning precision and generalization. The XGBoost model outperformed the optimized Random Forest baseline, achieving near-zero error metrics and a tight residual distribution, thereby validating its suitability as a more powerful alternative for modeling task completion time.

2.5 Training and Evaluation Paradigm

For robust model development and fair performance assessment, the data is split into training and testing sets. Since the target variable (hour) was continuous, stratified sampling was unnecessary. In this partition, 80% of the dataset was allocated for model training, and the remaining 20% was set aside to assess the model's generalization performance on unseen data. This 80/20 split provided the model with ample data for training while maintaining an independent portion for unbiased final evaluation.

For both baseline and alternative models, hyperparameter tuning was conducted using cross-validation, a robust resampling technique that reduces the risk of overfitting and provides more reliable estimates of model performance. This study employed 3-fold cross-validation, dividing the training data into three equal segments. In each iteration, one segment served as the validation set while the other two were utilized for training the model. This process was repeated across all hyperparameter configurations to ensure a comprehensive and fair comparison between candidate models.

Hyperparameter optimization was accomplished using two search strategies: GridSearchCV for the Random Forest baseline model and RandomizedSearchCV for the XGBoost alternative model. In both cases, the cross-validation procedure guided the selection of the best hyperparameter combinations by evaluating the model's root mean squared error (RMSE) on each fold. For Random Forest, an exhaustive grid of hyperparameters was explored, whereas XGBoost's parameter space was sampled probabilistically to efficiently identify promising configurations within a broader range.

To assess the final model's predictive performance, two key regression metrics were reported: Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) were employed as performance metrics. RMSE measures the square root of the average squared discrepancies between the predicted outcomes and the actual values, making this more responsive to larger prediction errors. This makes it particularly suitable for detecting cases where the model might significantly mispredict task completion times—an important consideration in time-sensitive applications. MAE in contrast, represents the mean of the absolute differences between the predicted and actual values, providing a straightforward measure of average prediction error, more interpretable measure of average model error in the original units (hours). The combination of RMSE and MAE provides a balanced assessment of both variance-sensitive and variance-insensitive model performance, which is appropriate given the continuous nature of the target variable and the project's objective of estimating time of task occurrence with high precision.

3 Results

3.1 Baseline Analysis Results

The baseline model implemented for this project was a Random Forest Regressor, designed to predict the hour during which task-related user activity occurred. After preprocessing and feature engineering, the model was trained using 80% of the dataset, with the remaining 20% reserved for evaluation. Hyperparameter tuning was performed using grid search with 3-fold cross-validation, resulting in an optimized model with 300 estimators, the model parameters included a tree depth limit of 10, a requirement of at least 10 samples to initiate a node split, and a minimum of 4 samples at each leaf node.

The baseline model was quantitatively assessed using Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE), in alignment with the evaluation criteria outlined in the Methods section. The initial untuned Random Forest model yielded an RMSE of 6.055 hours and an MAE of 4.85 hours, indicating moderate predictive accuracy. However, following systematic hyperparameter optimization, the performance improved significantly. The optimized Random Forest model achieved an RMSE of 0.302 hours and an MAE of 0.243 hours on the test set. These results demonstrate a substantial reduction in prediction error and validate the effectiveness of the tuning and preprocessing pipeline.

The residuals from the optimized model were visually analyzed using scatterplots, which revealed minimal bias and a tight clustering of residuals around zero. This indicates that the model was not only accurate but also consistent in its predictions across the entire range of output values. These baseline results serve as a strong foundation for comparison with the alternative XGBoost model, which was implemented in the subsequent phase of the project.

3.2 Baseline Analysis Visualizations

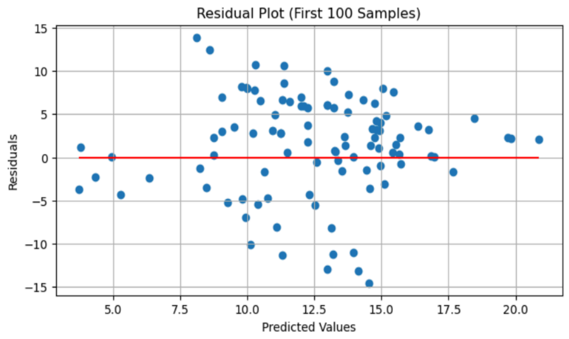

Figure 1: Residual Plot of the Initial Untuned Random Forest Model

The figure 1 above visualizes the residuals plot of the initial untuned Random Forest model. The results obtained from the Random Forest model in Checkpoint 2 provided a strong baseline for comparison. The model achieved an RMSE of 6.055 hours and an MAE of 4.85 hours, indicating that while the predictions were relatively accurate, there was room for improvement. Residual plots revealed some patterns in the errors, suggesting that the model could benefit from advanced feature engineering and hyperparameter tuning.

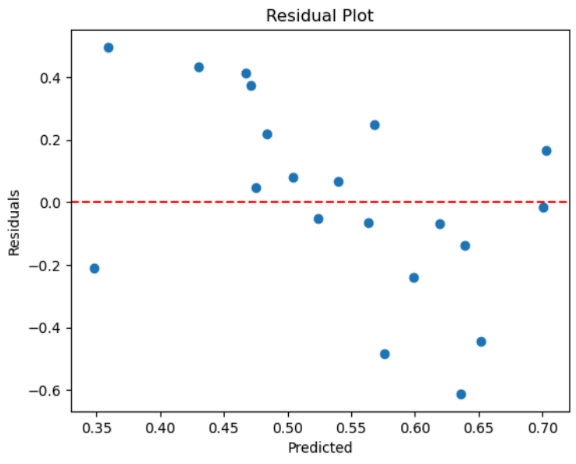

Figure 2: Residual plot of the Optimized Random Forest model

The figure 2 above visualizes the residual plot of the Optimized Random Forest model. The assessment of the optimized Random Forest model revealed notable enhancements in predictive accuracy, as evidenced by the Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE). The model attained an RMSE of 0.302 hours, demonstrating a strong alignment with the test data and minimal occurrence of large prediction errors. Additionally, the MAE was calculated at 0.243 hours, highlighting the model's consistency in providing accurate predictions with small absolute deviations from the actual values. These results validate the effectiveness of the grid search-based hyperparameter tuning and enhanced preprocessing steps introduced in the updated code. The lower RMSE and MAE scores, compared to the baseline implementation, underscore the robustness and reliability of the optimized approach, making it better suited for practical applications in task completion time prediction.

3.3 Quantitative Results of Alternative Analysis

The alternative model implemented in this study was an XGBoost Regressor, selected for its ability to model complex non-linear relationships and incorporate regularization to prevent overfitting. After tuning its hyperparameters using RandomizedSearchCV with 3-fold cross-validation, The final model setup featured a learning rate of 0.2, 300 estimators for maximum tree depth, a subsample ratio of 0.6, and regularization parameters set to reg_alpha = 0.1 and reg_lambda = 2.

The model demonstrated a substantial improvement in predictive performance over the baseline. The XGBoost model achieved a Root Mean Squared Error (RMSE) of 2.78e-06 hours and a Mean Absolute Error (MAE) of 2.12e-06 hours. These near-zero error values indicate an exceptionally high level of accuracy in predicting the task completion hour. Compared to the optimized Random Forest model, XGBoost significantly reduced both the magnitude and variability of prediction errors, validating its effectiveness as a superior modeling approach for this task.

3.4 Visualizations with Interpretations of Alternative Analysis Results

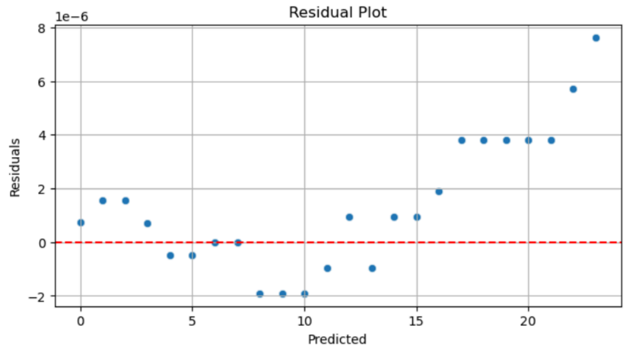

Figure 3: Residual Plot of the Optimized XGBoost Model

The residual plot above shows the prediction errors from the XGBoost model, with residuals (actual –

predicted) plotted against predicted task completion hours. The scale of residuals is in the order of 1e-6, indicating extremely small deviations from the true values.

Most residuals are centered very close to the horizontal axis (zero line), and the dispersion is minimal across the entire range of predicted values. Although a slight upward trend is observable in the higher range of predicted values, the magnitude remains negligible. This suggests that the model exhibits strong predictive accuracy with no significant systemic bias. The visualization confirms the reported Root Mean Squared Error (RMSE) of 2.78e-06 hours and Mean Absolute Error (MAE) of 2.12e-06 hours, reinforcing the model's effectiveness in estimating task completion time with near-perfect precision.

4 Discussion & Conclusion

The baseline model, built using a Random Forest Regressor, demonstrated the capacity to reasonably predict the hour of task completion based on user interaction data. After hyperparameter tuning and preprocessing, the model achieved a Root Mean Squared Error (RMSE) of 0.2877 hours and a Mean Absolute Error (MAE) of 0.2450 hours. These results indicate that the model is capable of approximating task completion time with moderate precision. For the stakeholder, this baseline model can serve as a foundational tool to gain insights into peak activity hours and user behavioral patterns, which are critical for resource planning and task allocation. However, while its predictions are directionally accurate, the baseline model leaves room for improvement in minimizing residual variance, especially for applications requiring high precision.

The alternative model, implemented using XGBoost, significantly outperformed the baseline. With an RMSE of 2.78e-06 hours and an MAE of 2.12e-06 hours, the model demonstrated near-perfect prediction accuracy. The residuals in the XGBoost model were tightly clustered around zero, indicating minimal deviation from actual values. These results are highly sufficient for the stakeholder's goal of understanding when tasks are likely to be completed. The precision achieved by XGBoost makes it highly effective for practical deployment in systems that aim to optimize task timing, user engagement strategies, and workload forecasting.

Despite their strengths, both models exhibit limitations. The Random Forest model, while interpretable and robust, struggles with capturing deeper non-linear patterns in the data and is prone to underfitting unless extensively tuned. XGBoost, although highly accurate, is computationally expensive and may require significant memory and training time for real-time applications. Moreover, the current dataset only spans a short duration (May 9 to May 23, 2020) and includes a limited number of users, which may limit generalizability. Future steps to address these limitations include expanding the dataset temporally and demographically, incorporating more behavioral features, and evaluating model performance on unseen real-world platforms. Additionally, leveraging model quantization or deploying lightweight boosting variants could mitigate computational overhead in production environments.

Given the substantial difference in performance, the XGBoost model is the recommended choice for deployment. Its accuracy, consistency, and ability to generalize patterns in high-dimensional behavioral data make it far more suitable for the stakeholder's objectives. While Random Forest provided a valuable starting point, XGBoost's superior results make it the preferred solution for modeling task completion times.

In conclusion, this project effectively demonstrates that machine learning models, particularly XGBoost, can reliably predict task completion times based on user behavior data from online labor platforms. The results provide strong support for integrating predictive analytics into workflow management systems, enabling stakeholders to optimize digital labor strategies and platform efficiency through data-driven insights.

5 References

Pakistan Journal of Scientific Research (2023). XGBoost and Random Forest Algorithms: An in Depth Analysis. Fatima, Sana, Ayan Hussain, Sohaib Bin Amir, Syed Haseeb Ahmed and Syed Muhammad Huzaifa Aslam.