Linking Writing Process to Writing Quality

Using typing behavior to predict essay quality

Matt Burns & Linhao Wu **(Apple_Juiced)**

Understanding the Writing Process: A Data Science Approach

- **Introduction to the Writing Process:**
  - Complex blend of behavioral actions and cognitive activities.
  - Various techniques in planning, revising, and strategic time allocation.
  - Influence of small actions on writing quality.
- **Limitations of Traditional Writing Assessments:**
  - Focus predominantly on the final product.
  - Lack of attention to the nuanced writing process.

Purpose of the Competition

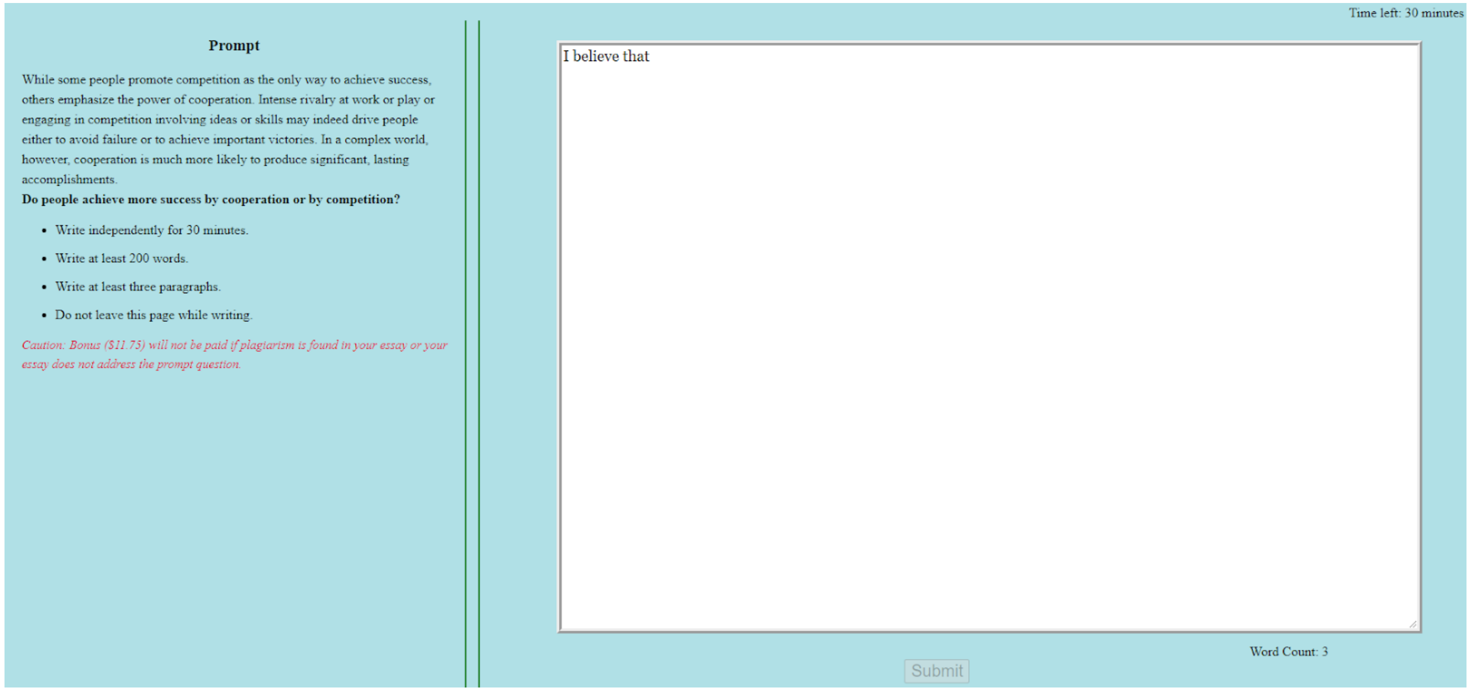
- **Innovative Research and Data Science:**
  - Exploring process features like pausing, additions, deletions, and revisions.
  - Utilization of large datasets for comprehensive analysis.
- **Competition Objective:**
  - Use keystroke log data to predict overall writing quality.
  - Identify correlations between writing behaviors and writing performance.
- **Impact on Learning and Writing Assessment:**
  - Shift from product-centric to process-centric assessment.
  - Enhancing learner autonomy, metacognitive awareness, and self-regulation in writing.

Keystroke Data Collection Procedure

- **Introduction:**
  - Participants hired from Amazon Mechanical Turk.
  - Required to use computers with keyboards.
- **Tasks Completed by Participants:**
  - Demographic survey, typing tests, argumentative writing task, vocabulary test.
- **Focus on Writing Task:**
  - 30-minute argumentative essay based on SAT prompts.
  - Random assignment of four different SAT-based prompts to control for prompt effects.

Participant Interface and Writing Prompt

- **Prompts**
  - Four prompts adapted from retired SAT prompts.
  - To control for prompt effect, participants were randomly assigned one of four prompts.
- **Instructions and Rules:**
  - Write independently for 30 minutes.
  - Minimum of 200 words and three paragraphs.
  - Participants need to stay on the page during the writing process.



Time left: 30 minutes

Prompt

While some people promote competition as the only way to achieve success, others emphasize the power of cooperation. Intense rivalry at work or play or engaging in competition involving ideas or skills may indeed drive people either to avoid failure or to achieve important victories. In a complex world, however, cooperation is much more likely to produce significant, lasting accomplishments.
Do people achieve more success by cooperation or by competition?

- Write independently for 30 minutes.
- Write at least 200 words.
- Write at least three paragraphs.
- Do not leave this page while writing.

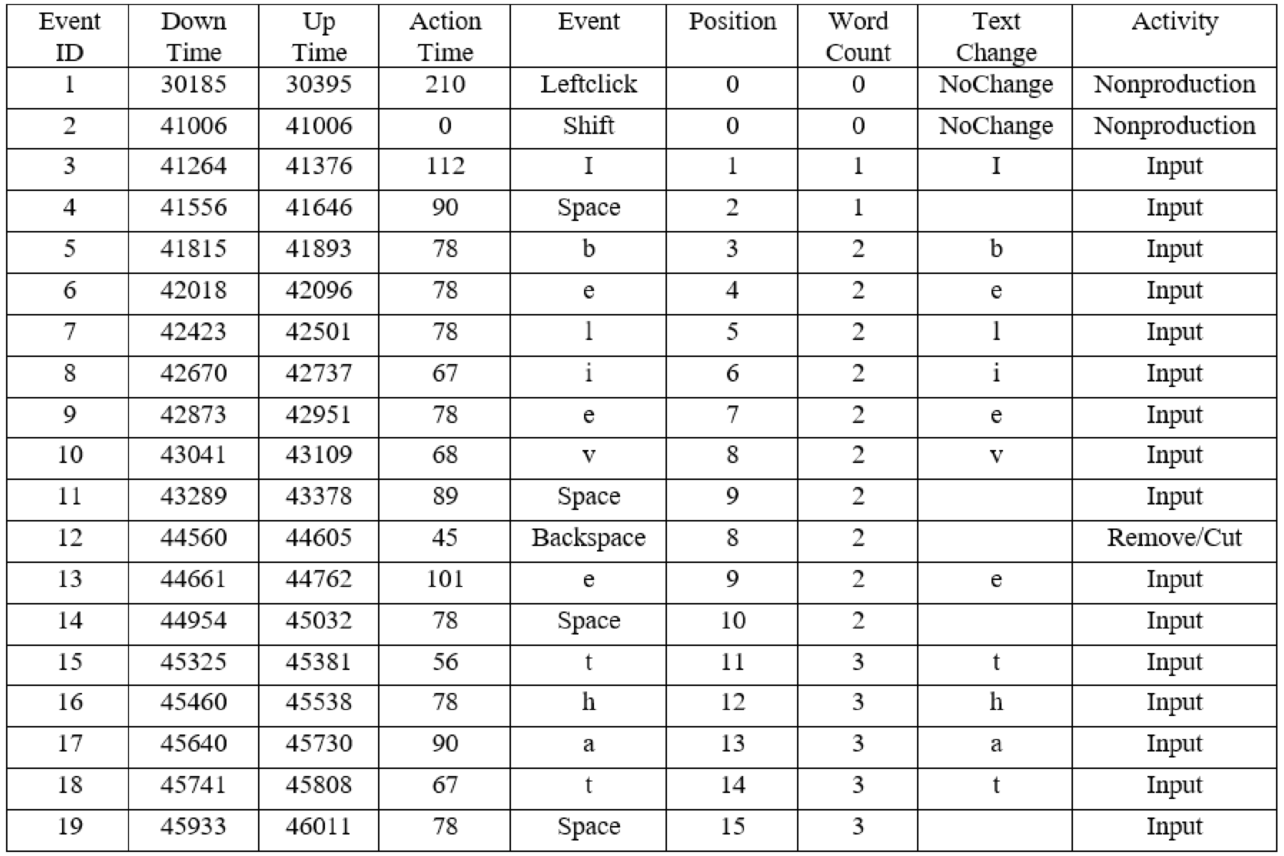
Caution: Bonus ($11.75) will not be paid if plagiarism is found in your essay or your essay does not address the prompt question.

I believe that

Word Count: 3

Keystroke Logging and Data Aggregation

- **Keystroke Logging Program:**
  - Developed in vanilla JavaScript, embedded in the project website.
  - Tracked keystrokes and mouse events, with time stamps and cursor positions.
- **Data Aggregation:**
  - Identification of operation types (input, delete, paste, replace).
  - Real-time reporting of text changes.



Event ID	Down Time	Up Time	Action Time	Event	Position	Word Count	Text Change	Activity
1	30185	30395	210	Leftclick	0	0	NoChange	Nonproduction
2	41006	41006	0	Shift	0	0	NoChange	Nonproduction
3	41264	41376	112	I	1	1	I	Input
4	41556	41646	90	Space	2	1		Input
5	41815	41893	78	b	3	2	b	Input
6	42018	42096	78	e	4	2	e	Input
7	42423	42501	78	l	5	2	l	Input
8	42670	42737	67	i	6	2	i	Input
9	42873	42951	78	e	7	2	e	Input
10	43041	43109	68	v	8	2	v	Input
11	43289	43378	89	Space	9	2		Input
12	44560	44605	45	Backspace	8	2		Remove/Cut
13	44661	44762	101	e	9	2	e	Input
14	44954	45032	78	Space	10	2		Input
15	45325	45381	56	t	11	3	t	Input
16	45460	45538	78	h	12	3	h	Input
17	45640	45730	90	a	13	3	a	Input
18	45741	45808	67	t	14	3	t	Input
19	45933	46011	78	Space	15	3		Input

Extracting Insights: Feature Extraction Process

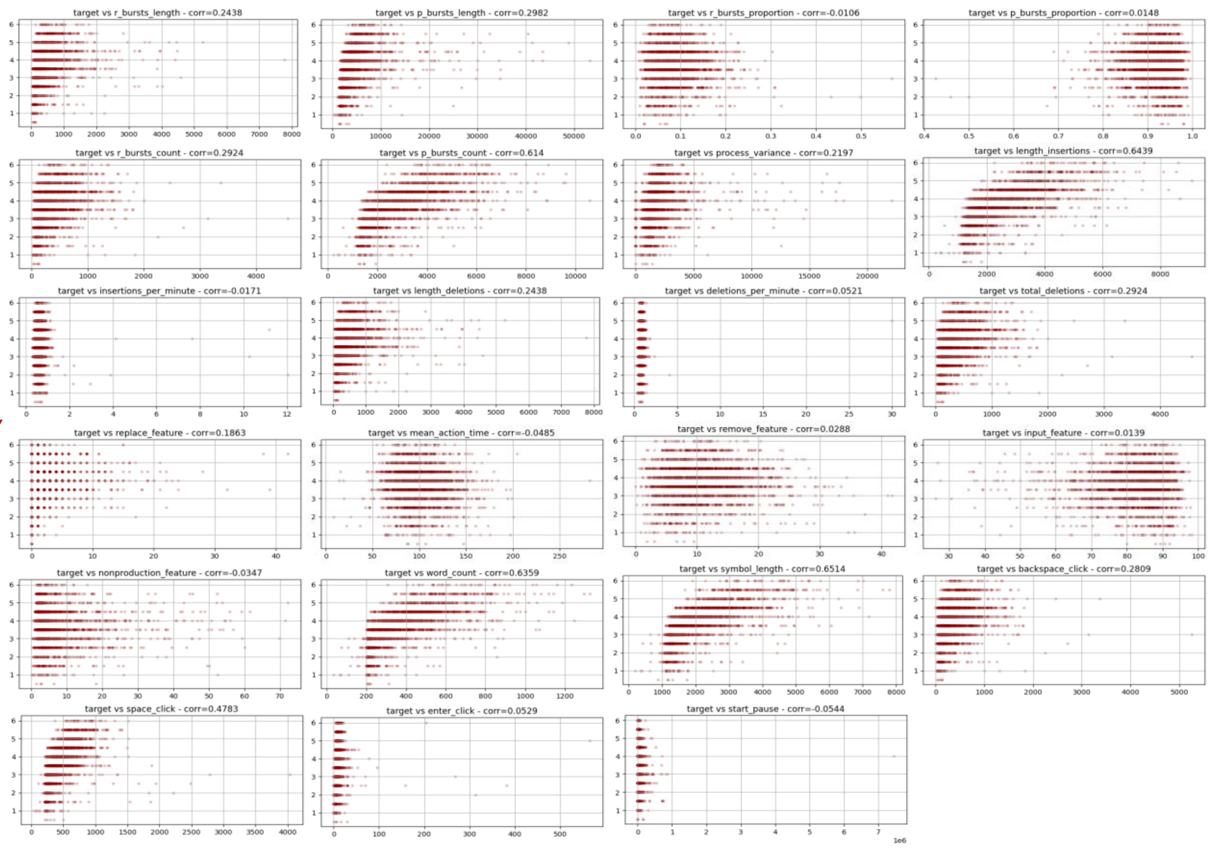
- The creators of this competition provided a set of recommended feature extractions, that were common for keylogging datasets.
- We utilized this list and extracted all the recommended features.
- Our initial models utilized all these features. We utilized selection techniques to differentiate between features with significant predictive value and those that were not correlated with higher or lower scores.

start_pause	*enter_click*	*space_click*	backspace_click
symbol_length	text_length	*nonproduction_features*	input_features
remove_features	mean_action_time	replace_features	sentence_size_features
total_deletions	*deletions_per_minute*	*insertions_per_minute*	*length_insertions*
process_variation	*p_bursts_count*	r_bursts_count	p_bursts_proportion
r_bursts_proportion	*p_bursts_length*	*r_bursts_length*	*word_count*
tch_unique	*summary_time*	*number_sentence*	

Features extracted from dataset, features in *italics* are included in our regression models, see supplement for description of each extracted feature.

Feature Selection and Correlation

- We charted our extracted features against score, we initially selected features with high correlation and visual linearity.
- Charts with low linear correlation tend to have less predictive value.
- Charts with higher linear correlation tend to be more valuable as predictive variables.

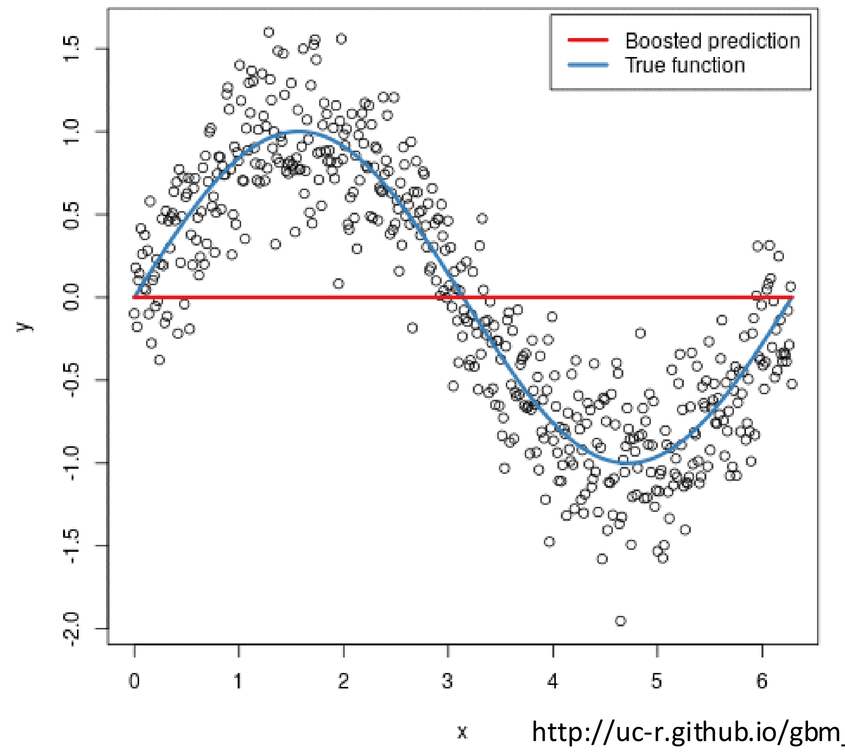


Correlation plots for selected features.

Model Design

- We examined previous work on similar datasets, as well as the dataset in question.
- Reviewing approaches to previous datasets we found some commonalities.
  - GBMs (Gradient Boosting Machines)
  - SVMs (Support Vector Machines)
  - Neural networks

- Having examined the approaches of other submissions for this specific dataset we determined that GBMs were a promising option.



http://uc-r.github.io/gbm_regression

Leveraging Proven Models

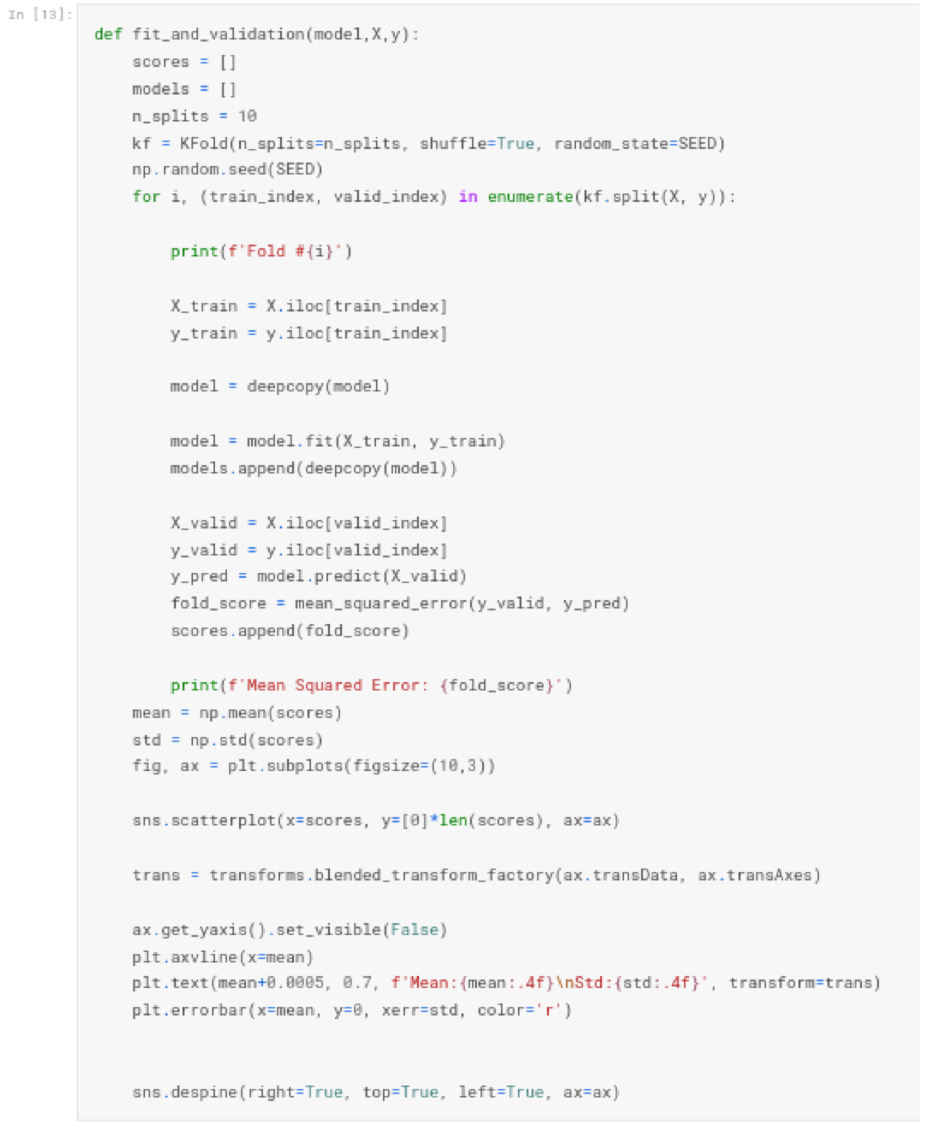
- **XGBoost:** Exceptional tuning capabilities with access to numerous parameters for optimizing precision and accuracy.
- **CatBoost:** Specialized in dealing with categorical features without extensive pre-processing, making it efficient and user-friendly.
- **RandomForest:** Easy to use and interpret, providing stable and reliable results across various datasets.

Random forest

dmlc XGBoost

Model Selection and Implementation

- To simplify the testing process, we utilized a wrapper function for our models.
- This provided standardized input and output. Allowing for easy comparison.

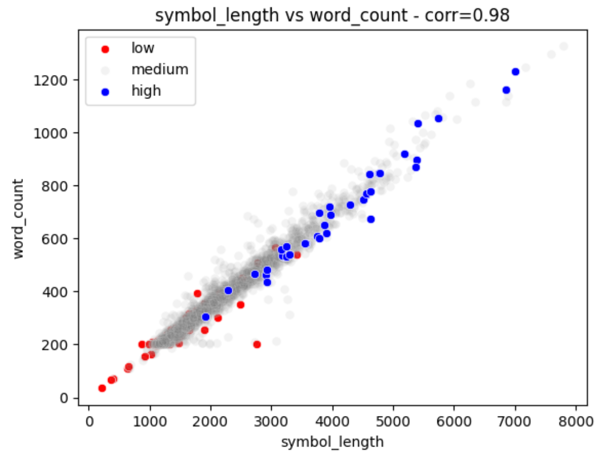


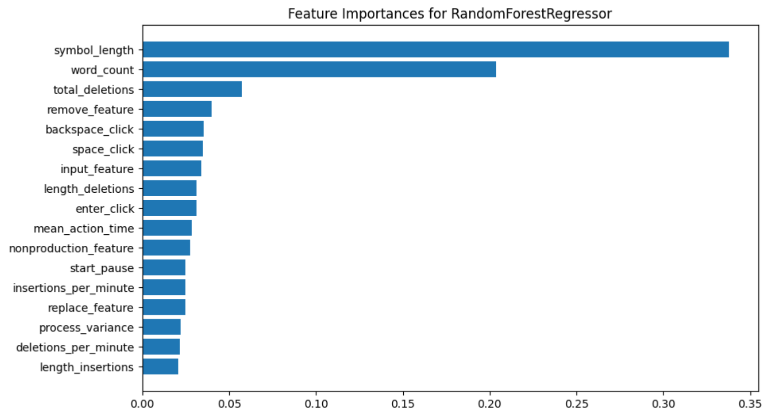
```
In [13]:
def fit_and_validation(model,X,y):
    scores = []
    models = []
    n_splits = 10
    kf = KFold(n_splits=n_splits, shuffle=True, random_state=SEED)
    np.random.seed(SEED)
    for i, (train_index, valid_index) in enumerate(kf.split(X, y)):

        print(f'Fold #{i}')

        X_train = X.iloc[train_index]
        y_train = y.iloc[train_index]

        model = deepcopy(model)

        model = model.fit(X_train, y_train)
        models.append(deepcopy(model))

        X_valid = X.iloc[valid_index]
        y_valid = y.iloc[valid_index]
        y_pred = model.predict(X_valid)
        fold_score = mean_squared_error(y_valid, y_pred)
        scores.append(fold_score)

        print(f'Mean Squared Error: {fold_score}')
    mean = np.mean(scores)
    std = np.std(scores)
    fig, ax = plt.subplots(figsize=(10,3))

    sns.scatterplot(x=scores, y=[0]*len(scores), ax=ax)

    trans = transforms.blended_transform_factory(ax.transData, ax.transAxes)

    ax.get_yaxis().set_visible(False)
    plt.axvline(x=mean)
    plt.text(mean+0.0005, 0.7, f'Mean:{mean:.4f}\nStd:{std:.4f}', transform=trans)
    plt.errorbar(x=mean, y=0, xerr=std, color='r')


    sns.despine(right=True, top=True, left=True, ax=ax)
```

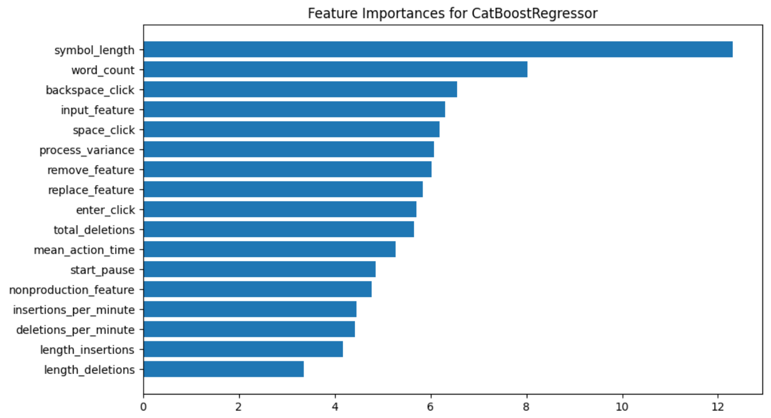
Refining Model Performance: Comparing Feature Weights

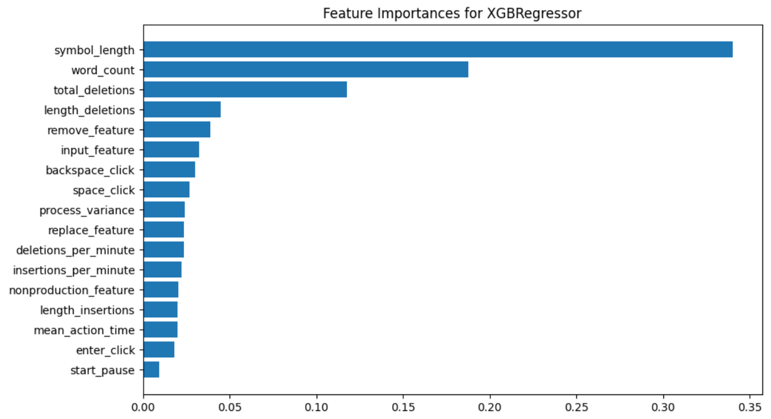
- Feature Weights: : Feature weights in machine learning models represent the importance or influence each feature has in predicting the target variable. In models like XGBoost, CatBoost, and RandomForest, these weights help us understand which aspects of the data are most critical for making accurate predictions.
- Examining these charts, we made refinements to our model by eliminating features that weren't contributing as significantly to model prediction.
- This improved performance by reducing overfitting of the model data.

- Consistently, *symbol_length* and *word_count* were the most strongly weighted features in the compared models.
- Including both these features improved predictions, eliminating either or both had a negative effect.



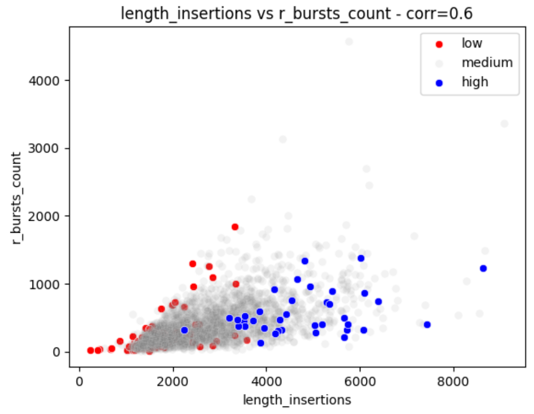


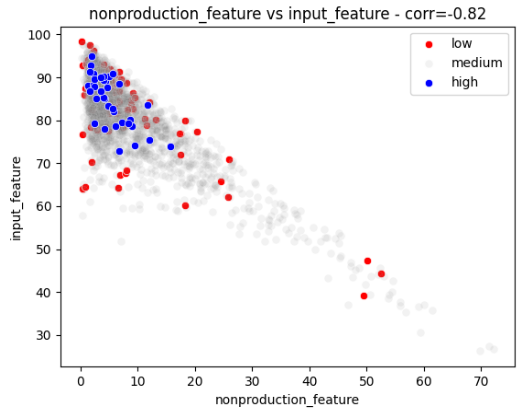




Improving Model Performance: Multiple Approaches

- We explored other methods that we could utilize to improve our model performance over the RandomForest baseline
- After exploring options to improve performance we settled on further feature engineering, ensemble techniques, and outlier elimination as potential avenues for improvement.

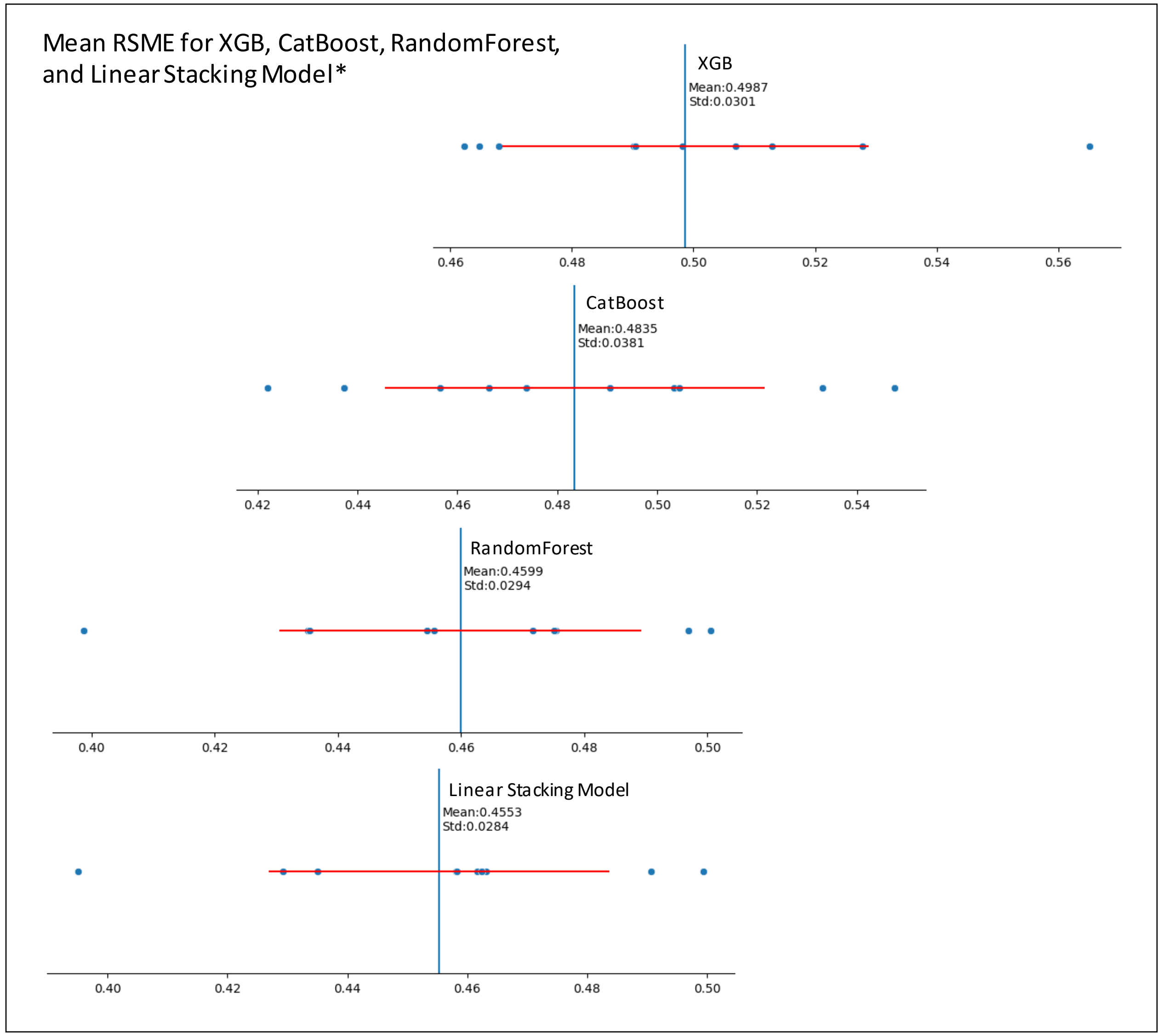




- Further feature engineering did improve feature performance for our RandomForest model and allowed us to further reduce the number of features utilized, while improving predictive performance.
- Outlier elimination, at least following the method that we utilized, significantly decreased performance. We suspect that out elimination was too aggressive, and a more nuanced approach could be successful.
- Ensemble was also successful at improving our performance.

Ensemble Techniques

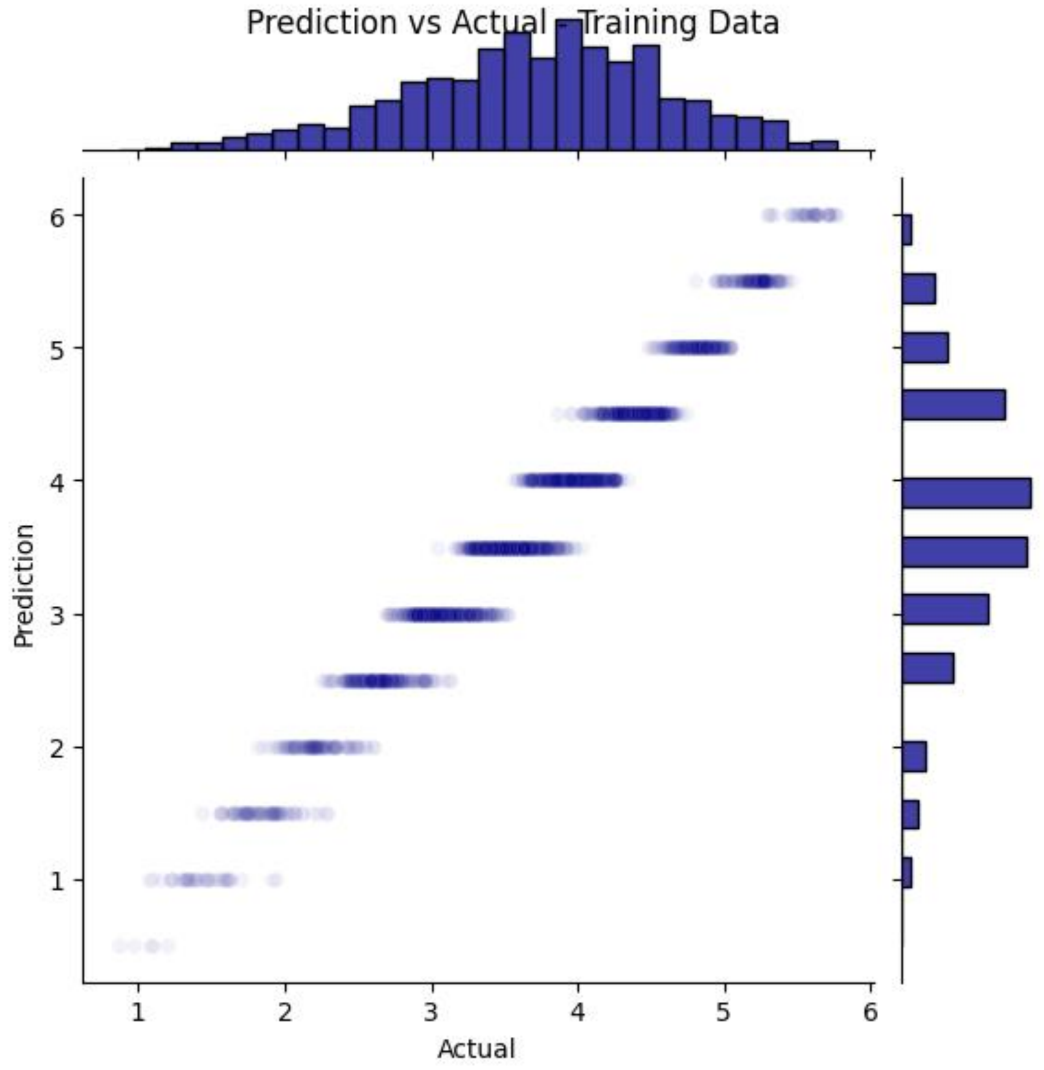
- We utilized a simple linear stacking model with all three our GBM models.
- The advantage of this approach is that you can leverage the predictive ability of all three models, while avoiding some of the disadvantages of all three.
- Our linear stacking model achieved better predictive performance than any of the other three models we tested on their own.



*Lower mean RSME indicates better model performance.

Future Improvements

- Further base model tuning
- Composite feature engineering
- Ensemble with more complex secondary models



Conclusions and Takeaways

- Building models that can accurately predict connections during the process of revision, and the quality of the final product is difficult.
    - One of the more predictive features (word_count) isn't directly connected to revision.
- The difference between our model and the best performing model in this competition in terms of RSME score is only 0.076.