Part 1. Introduction

In this project, we focus on developing an Ordinal Regression model to classify the levels of cognitive depth elicited by questions from respondents. Our dataset consists of randomly sampled questions from analysts in financial conference calls, making our data domain-specific. This model aims to predict the complexity of the thinking required to answer these questions, a process which is inherently intricate.

Part 2. Data

We performed a random split, the table below shows the distribution of the cognitive levels

	1	2	3	4	Total
Train	52	102	98	48	300
Dev	23	32	30	15	100
Test	11	48	22	19	100
Total	86	182	150	82	500

- Level 1 accounts for 17.2% of the total dataset, with 10.4%, 4.6%, and 2.2% in the train, dev, and test sets, respectively.
- Level 2 constitutes the majority at 36.4% overall, divided into 20.4% in training, 6.4% in dev, and 9.6% in the test set.
- Level 3 has 30% of the total data, with 19.6% in the train set, 6% in dev, and 4.4% in testing.
- Level 4 makes up the remaining 16.4% of the data, with 9.6%, 3%, and 3.8% in the train, dev, and test sets, respectively.

The data is not perfectly balanced, but it aligns with our expectation considering the nature of the labels. Labels 1 and 4 represent more extreme values, which are naturally less common than the moderate labels 2 and 3. Most of the questions posed by analysts are categorized as Level 2 or Level 3, indicating that they often seek to clarify information, inquire about management methodologies, or make informed inferences during the Q&A sessions. On the other hand, analysts spend less time on straightforward factual queries or on soliciting complex analyses.

Part 3. Feature Construction

1. Bag of Words (BoW): We implement variations of the BoW.

- **question_word_count**: A count of specific words in a *question word list.* It can help identify patterns or common themes in the questions.

- **bow_featurize**: A frequency-based BoW where each word is represented by its frequency of occurrence in the text. From this, we can measure the prominence of certain terms within a text, which can indicate its focus or relevance to specific topics.
- **TF-IDF**: Similar to BoW but also take the inverse frequency across questions into account.
- **bigram**: A frequency-based BoW that looks at two tokens at a time. Examining pairs of words allows us to capture phrase-level nuances and common collocations that single-word analysis might miss.

2. **Syntactic Features:** These features assess the structure of text.
    - **get_length**: Counts the number of tokens in a question. Because the length of a text can suggest its complexity or detail level.
    - **calculate_syntactic_complexity**: Measures the depth of the parse tree in sentences using NLP library spacy, indicating their grammatical intricacy. A deeper parse tree indicates more complex sentence structures, which can reflect higher informational or linguistic complexity.
    - **complexity_readability_feature:** Calculates different types of readability scores (automated_readability_index, coleman_liau_index, flesch_readability_ease, gulpease_index, gunning_fog, lix) for each sentence in a dataset and normalizes these scores across sentences. The scores could provide insights into the questions' complexity and readability which may be relevant for their cognitive level.
3. **Sentiment Analysis Features:** These features help gauge the emotional tone of the questions.
    - **Afinn_sentiment** and **Vader_sentiment**: Assigns sentiment classifications or scores based on the Afinn or Vader lexicon. These features assess the emotional tone of the content, and a deeper question may have a more emotional tone.
    - **finbert_tone**: Tone score from finBERT, a BERT model that was trained on financial data. It can highlight the underlying sentiment or tone specific to financial contexts, important for financial data analysis. Similar to the previous feature, a deeper question commonly has a more emotional tone.
4. **Part of Speech (POS):** These features tackle the underlying linguistic form of the sentence
    - **vb_jj_pos_tag:** Counts of the various verb and adjective forms of POS-tags**.** From this, we can analyze the question from a linguistic point of view.
    - **liwc_pos_type (Linguistic Inquiry and Word Count):** LIWC analyzes word counts based on psychological and linguistic categories, such as pronouns and emotion-related words.
5. **LLM Contextual Embedding:** Utilize embedding as a representation of the question
    - **roberta_emb_func**: Embeddings given by roBERTa that may capture unobservable sentence-level context. The roBERTa embeddings can capture the subtle contextual clues within a sentence. Therefore, it can give an informative understanding.

Part 4. EDA

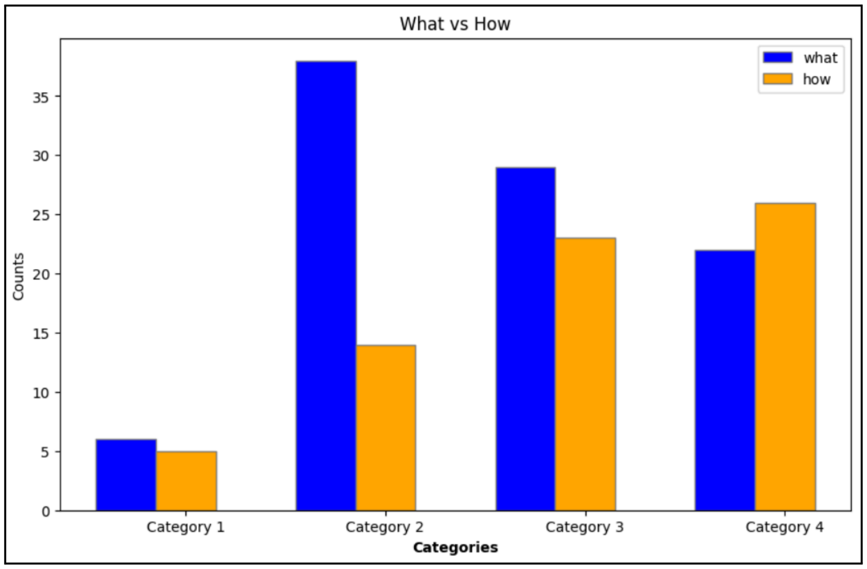


Fig 1: Distribution of the count of “what” and “how” at each level

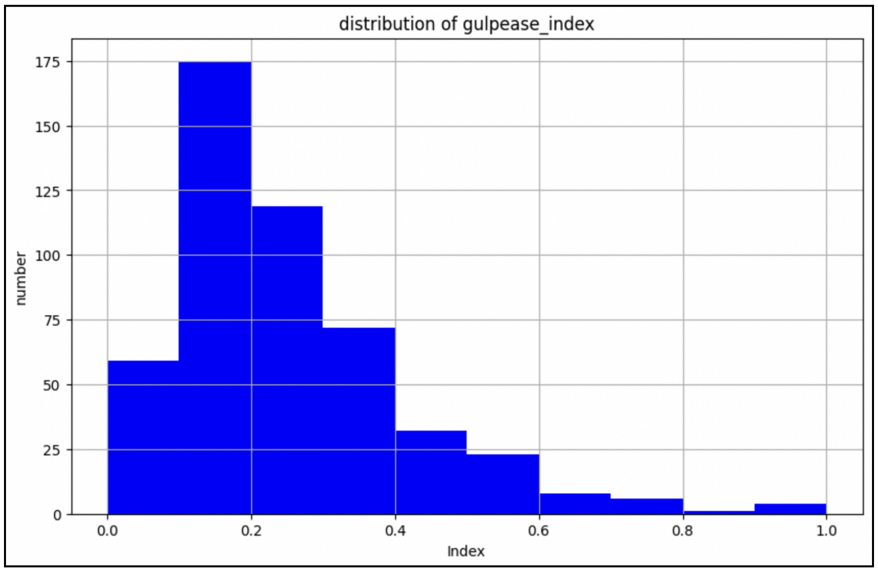
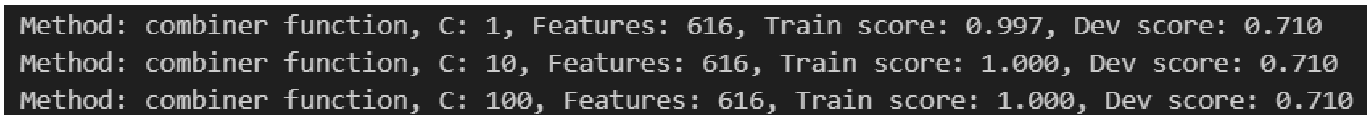


Fig 2: Distribution of gulpease index (from the complexity_readability_feature)

Within question_word_count, phrases like “What” and “Why” are likely significant indicators of cognitive level, and Figure 1 provides two instances in such question words. Observe that “What” is more prevalent in lower-level questions and “How” is more common in higher-level questions. From Figure 2, we can see that the distribution of the gulpease index is not normal and it is right-tailed. The skewness shows that most of the questions are easy to read (low score) and only a small number of questions are hard to read which may indicate that they have a high cognitive level and large thinking depth.

Part 5. Modeling

Given that our outcome variable ranges from level 1 to 4 with an inherent order, we choose to build on an Ordinal Regression model. During our experimentation, we observed that incorporating the Bow_featurize substantially improved the model's predictive capability. The addition of other features did not yield any enhancement and sometimes even degraded it when combined with Bow_featurize. Nevertheless, this strategy is not advisable as it clearly leads to overfitting as it reaches an accuracy of 1 on the train data.



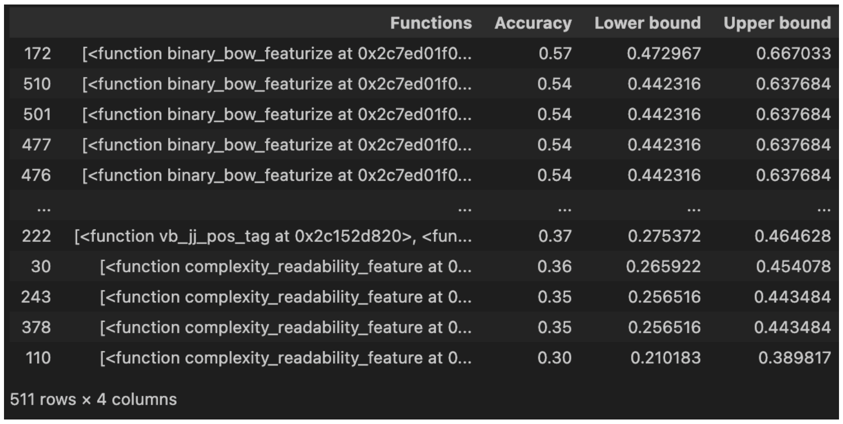
```
Method: combiner function, C: 1, Features: 616, Train score: 0.997, Dev score: 0.710
Method: combiner function, C: 10, Features: 616, Train score: 1.000, Dev score: 0.710
Method: combiner function, C: 100, Features: 616, Train score: 1.000, Dev score: 0.710
```

Fig 2: Overfit with Bow_featurize

After exploring distinct feature combinations, we decided to select the final features out of the following set, which covers all aspects discussed above:

- binary_bow_featurize
- bigram
- vb_jj_pos_tag
- complexity_readability_feature
- get_length
- roberta_emb_func
- finbert_tone
- question_word_count
- calculate_syntactic_complexity

Using the nine features, we run the model across all possible feature combinations and compare their performance, iteratively identifying the optimal model from the $2^9 - 1 = 511$ distinct configurations (the model with no covariates is ignored). The final ranking of our models based on accuracy is:



	Functions	Accuracy	Lower bound	Upper bound
172	[<function binary_bow_featurize at 0x2c7ed01f0...	0.57	0.472967	0.667033
510	[<function binary_bow_featurize at 0x2c7ed01f0...	0.54	0.442316	0.637684
501	[<function binary_bow_featurize at 0x2c7ed01f0...	0.54	0.442316	0.637684
477	[<function binary_bow_featurize at 0x2c7ed01f0...	0.54	0.442316	0.637684
476	[<function binary_bow_featurize at 0x2c7ed01f0...	0.54	0.442316	0.637684
...	...	...	...	...
222	[<function vb_jj_pos_tag at 0x2c152d820>, <fun...	0.37	0.275372	0.464628
30	[<function complexity_readability_feature at 0...	0.36	0.265922	0.454078
243	[<function complexity_readability_feature at 0...	0.35	0.256516	0.443484
378	[<function complexity_readability_feature at 0...	0.35	0.256516	0.443484
110	[<function complexity_readability_feature at 0...	0.30	0.210183	0.389817

511 rows × 4 columns

Fig 3: Table for Feature Selection

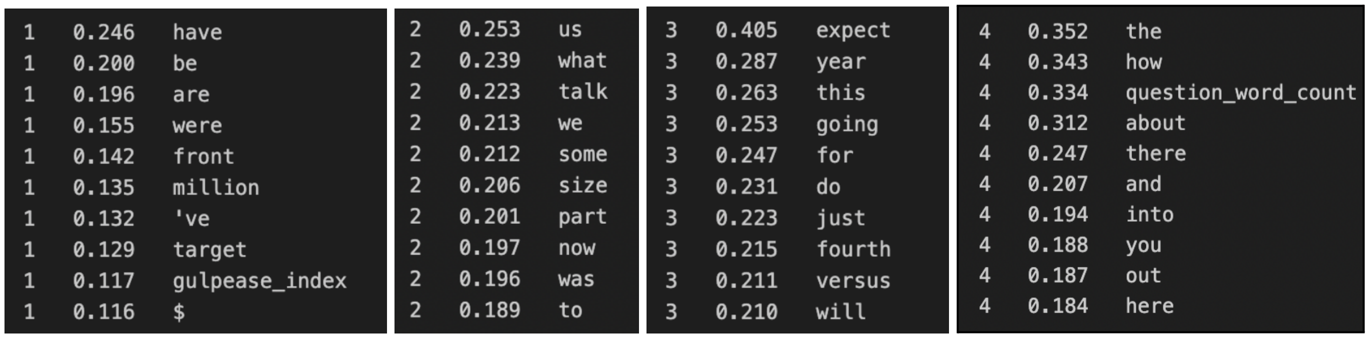
The optimal features are **binary_bow_featurize, complexity_readability_feature, finbert_tone,** and **question_word_count**.

Part 6. Hyperparameter Tuning

Using the optimal features identified during the modeling stage, we apply a **Grid Search** to Ordinal Regression by adjusting parameters: class weights (balanced or none), regularization method (with or without L2), and regularization strength (C values in the set $[0.1,\ 0.5,\ 1,\ 3,\ 5,\ 10,\ 20,\ 100]$). The hyperparameters that give the highest accuracy are "balanced" class_weight, "L2" regularization, and "C = 100" as regularization strength.

Part 7. Discussion

After experimenting with different features and tuning for the optimal hyperparameters, we found that the BoW-based features still dominate model performance. On one hand, it indicates that our annotation guidelines were on the right track by examining specific words. For instance, a question that contains the word "predict" is likely to be a question with a higher cognitive level. Intuitively, the basic indicators — 'who,' 'what,' 'where,' 'when,' 'why,' and 'how' — should offer valuable cues on identifying the emphasis of a query. Additionally, we discover that the complexity and readability of the question help in determining the cognitive level. Lastly, the final feature set includes sentiment analysis from finBERT. This is unexpected because previous combinations of Afinn and Vader sentiment fail to show significance.



```
1   0.246   have
1   0.200   be
1   0.196   are
1   0.155   were
1   0.142   front
1   0.135   million
1   0.132   've
1   0.129   target
1   0.117   gulpease_index
1   0.116   $
```

```
2   0.253   us
2   0.239   what
2   0.223   talk
2   0.213   we
2   0.212   some
2   0.206   size
2   0.201   part
2   0.197   now
2   0.196   was
2   0.189   to
```

```
3   0.405   expect
3   0.287   year
3   0.263   this
3   0.253   going
3   0.247   for
3   0.231   do
3   0.223   just
3   0.215   fourth
3   0.211   versus
3   0.210   will
```

```
4   0.352   the
4   0.343   how
4   0.334   question_word_count
4   0.312   about
4   0.247   there
4   0.207   and
4   0.194   into
4   0.188   you
4   0.187   out
4   0.184   here
```

Fig 4: Top significant features for each class

Furthermore, we implement a logistic regression using the same covariates to obtain feature importance for each cognitive level (Fig 4). The analysis revealed certain features that are intuitively aligned with the expected cognitive demands of each level. Notably, the term "expect" is the most defining feature for level 3, which is consistent with its frequent use in contexts requiring forecast or prediction, tasks that are inherently higher on the cognitive scale. Surprisingly, we discover tokens that show strong association with specific levels despite being previously overlooked. For example, "$" and "million" from level 1 are usually found when analysts ask for specific company financial metrics. This indicates that the question is likely to be fact-based and is less cognitively demanding. Overall, many of the significant features across all levels come from a simple BoW method, confirming its efficiency in our task.

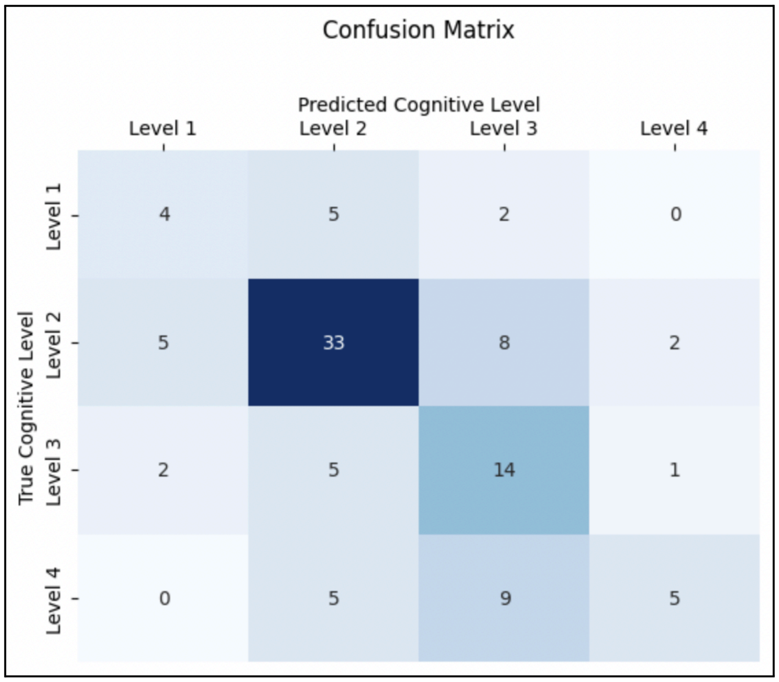


Fig 5: Confusion matrix for the test dataset

From the confusion matrix, we can see that the model correctly classifies 69% of the Level 2 questions, demonstrating good predictive abilities for the majority class. However, it shows confusion in classifying the other cognitive levels, particularly mistaking Level 1 questions for Level 2, and Level 3 questions for Levels 2 and 4. This could indicate the need for improved feature engineering and modeling techniques; however, we also encountered similar classification mistakes during the annotation process. Thus, it potentially calls for better clarity in our annotation guidelines, particularly regarding the boundaries of adjacent cognitive tags or levels.

Part 8. Future Work

- **Guideline**: Future researchers can expand on our dataset and extend to more general topics with potentially new question types. For instance, a speech act question like “Can you pass me a drink?” doesn’t match any of the categories given by our guideline.
- **Model**: Considerable effort has been dedicated to finding the optimal set of features and hyperparameters. Looking at the top covariates, we can observe that some miscellaneous words are significant in the model which can’t be the case. Therefore, finer feature selection on top of feature category selection can be performed. Additionally, our experiments were limited to using only Ordinal Regression as the underlying model. Potentially, exploring other models such as a variation of neural network may perform better with the same feature set after tuning.