What You Say and How You Say it Matters

An Exploration of Earnings Calls and Question Generation

Xiaofan Bai, Yiyan Chen, Isaac Haberman, Lu Yin
New York University

Contact Information:
Center Of Data Science
New York University
60 5th Ave, New York, NY 10011

Github Link:
https://github.com/NYU-CDS-Capstone-Project/NYU_Capstone_Data-Crusher

Abstract

To better prepare JP Morgan Chase's Investor Relations Team for upcoming earnings calls, we have developed a machine learning modeling framework to generate tag predictions for a given set of analysts. By transforming the analysts' questions into word vectors and utilizing cutting edge dimension reduction techniques, we achieved workable results and produced sample questions for each analyst and tag. With the framework, the Investor Relations Team will be able to mathematically judge their intuitions and better prepare the Chief Executive Officer (CEO) and Chief Financial Officer (CFO) for the earnings calls.

Introduction

Each quarter, public companies hold an earnings call, a call between the CEO and CFO of the company and well-known financial analysts. While the call is preceded by opening statements from the company, the question and answer session between the company and the analysts is the essence of the call. Each side has a goal for the call, with the analysts looking for insight into the company and the company trying to please the analysts. Given these goals, we sought to develop a machine learning model that would predict the types of questions the analysts would ask prior to the call. With this in hand, the JP Morgan Chase Investor Relations Team would be better prepared for earnings call.

Main Objectives

1. Exploratory Data Analysis: Explore the distribution of analysts and tags across each other, the four quarters of the year, the years themselves and the questions in our dataset.
2. Tag Prediction: Given a set of 14 tags, developed specifically for earnings calls, develop a machine learning model that provides predictions per analyst of each of the tags.
3. Question Generation: Using the tag predictions, generate sample questions for each analyst and predicted tag.

Materials and Methods

Our original dataset was a heavily marked-up transcription of 160 earnings calls and other analyst calls from major U.S. banks including Citigroup, JP Morgan Chase and Goldman Sachs. Each transcript was spread over N rows, with each row representing a question and answer. Each datum had the following additional features: date of the call, the company participants, the questioner, and the question tag. The tags were developed internally by JP Morgan Chase, for the explicit purpose of better classifying earnings call questions.

Results

Exploratory Data Analysis

Our initial exploratory analysis explored the bivariate distributions between the analysts and tags, the analysts and words, the tags and words and the tags and quarters. We noted a number of interesting results:

1. The distribution of tags across quarters varied heavily. While we had initially suspected this given the financial events that occur across specific times of the year, this served as confirmation of our suspicions.
2. Once we normalized the analysts per appearance, there were fewer differences in distributions among the tags. **Figure 1** shows a typical distribution of tags among one of the more popular analysts.
3. Using a TF-IDF transformation, there were considerable syntactical differences among the analysts and separately among the tags.

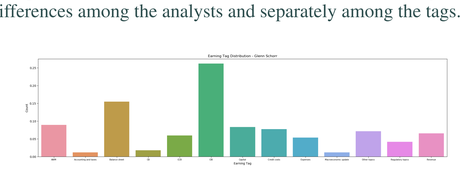

Figure 1: Distribution of tags by analyst

Tag Models

1. **Multi-Class Question Classifier**: A multi-class classifier predicted the tag for each question in the dataset. Overall, model performance was poor, with the model able to classify in the One vs. the Rest situation, but unable to distinguish between tags when tested over all tags.
2. **Tag Model V1**: Each analyst and tag combination was tested per analyst in a given earnings call. This model performed better than the Multi-Class Question Classifier, but suffered from its large dimensionality. Since each analyst and tag needed to be encoded in the model, the dataset was overly sparse and hard to predict on. To alleviate these concerns, we tested dimension reduction methods such as Principal Component Analysis and Non-negative Matrix Factorization. Even with the reduced dimensionality, model performance remained the same.
3. **Tag Model V2**: Since we had previously noted syntactic differences and similarities among analysts and among tags, we experimented with further dimension reduction on the analyst and tag features to reduce sparsity. Instead of representing each analyst as a one-hot encoded feature, each analyst would be represented as a normalized vector of weights. For both the analysts and tags, we designed new matrices *Analyst X Word* and *Tag X Word*. Instead of using the original questions, we used a BM-25 transformation along with common bigrams and trigrams. For the new matrices, we tested different implementations of Non-negative Matrix Factorization, ultimately reducing the analysts to a vector of 6 features, and the tags to a vector of 3 features. The new models proved to be vastly more successful, as shown in **Table 1**.

Model	AUC ROC	Accuracy
Multi-class Tag	.21	0.25
Tag V1	0.68	0.72
Tag V2	0.78	0.86

Table 1: Model performances for best iteration of each model.

Question Generation

We used multi-layer GRU and LSTM models to predict word probabilities in a sequence model. Due largely to the small available dataset, we were unable to generate relatively higher semantical questions. The sequence method of the Question Generation model using word-by-word sequence generation can be seen in **Figure 2**. Results of the Question Generation model using character-by-character sequence generation, the better performing model, can be seen in **Figure 3**. Due to the poor performance of the models, we developed the **Question Sampling** method.

```
sample #0: ...on net interest income , do you have an outlook -> 'net'
sample #1: ...net interest income , do you have an outlook for -> 'interest'
sample #2: ...interest income , do you have an outlook for how -> 'income'
sample #3: ...income , do you have an outlook for how the -> ','
sample #4: ..., do you have an outlook for how the net -> 'do'
```

Figure 2: Sample sequence from word-by-word sequence generation

```
[Can you give us ]an update on the share of your portfolio ?  Okay . And on the securi
[Can you give us ]an update on the short end ? And then just finally , I guess the cor
[Can you give us ]a sense of consumer loan growth ? I guess I guess I was wondering if
[Can you give us ]some color on the pace of the purchase accounting accretion and loar
[Can you give us ]a sense of your comments about the deposits , but you guys have beer
```

Figure 3: Sample Results from Character-By-Character Sequence Generation

Question Sampling

Given the poor performance of the **Question Generation**, we developed a question sampling method alongside **Tag Model V2**. We created a third matrix, *Question X Word*, which we subsequently used to find the cosine similarity between the analyst and tag matrices (separately). We treated the resulting values as affinity scores and averaged the results across the similarity matrices to find ideal sample questions per analysts and tags.

Conclusions

- The syntactical differences among the analysts allowed us to design models that used the text of the questions and not just the meta-data of the analyst and company.
- Without copious amounts of additional data, the **Question Generation** model will not produce satisfactory results.

Forthcoming Research

While we applied out methodology to the specific problem of earning calls at major U.S. banks, there are other professions in which better preparedness to questions would be helpful. We are interested in applying our methods to the medical profession, specifically to preparing medical professionals to anticipate patient questions. Similarly, teachers could use such a model to best prepare for class each day. Lastly, we hope to test our question generation models with additional data.

References

[1] Golub Brunet, Tamayo and Mesirov. Metagenes and molecular pattern discovery using matrix factorization. *Proceedings of the National Academy of Sciences of the USA*, 101(12), 2004.

[2] DeWilde. Textacy: Nlp tools Python, 2016–. [Online; accessed ¡today¿].

[3] Marinka Zitnik and Blaz Zupan. Nimfa: A python library for nonnegative matrix factorization. *Journal of Machine Learning Research*, 13:849–853, 2012.

Acknowledgements

We would like to thank Bruno Goncalves, Santiago Salazar, Fernando Cela Diaz, and Boyu Wu at JP Morgan Chase for serving as project mentors. We would also like to that Professor Richard Bonnau for his guidance.