1. soft skills: communication and collobration

Business analyst, consultants, listen before I judge/ explain the logic and articulate my idea/ express concerns without offending/; written documentations and comments / well and efficient face-to-face discussion

2. technical skills

CS fundamentals: data structer, algorithms, database, operating systems and distributed computing

Practical tools: angularJS, SQL, Linux Command Lines

Intership: 4000+ coding, test suits and maintainable codes, (MySQL, ElasticSearch and Git)

3. ML/natrual language processing

Theorical (CS + Statistics) and practice: Facebook paid-post detection, user-intention extraction, relation-extraction from government purchase documents.

I hope to start my career at IBM: one of the most influential cloud computing and artifical intelligence, to learn from the frointier of the industry and make impact to the large business world.

Most diffcult task:

1. Suprisingly no libraries to do: lack of data. not machine-learning solvable, have to look for hand-written rules.

2. Complexity: very tempted to write codes that will work for first but can’t generalize. Lost tracbility for later types of documents. Slow down and investigate more than 100 documents from different sources and conclude their general types. Then re-design the overall structure so that first step is to classify documents into each type and then call the fucntion specifically for that type.

3. Maintainable code: similar to deal with complexity, when the code is modulized. It’s much easier to understand and maintain. But still, I need to figure out a way to allow new coders to understand all the rules so I need to write many examples and put the. Simplify several levels of if-else into one or two levels by introducing more variables describing

4. Fast code: classifiying document types need to recogniztion the entities that need to upload the file to a webserver and get recognition results. Later on, a document need to be interpretated and use recognition service again. No to mention sometimes a document might just contain an other document inside a table cell. So the code need to be recursive to parse those documents inside documents. However, that increase the use of network. 95% time spend is to wait for the network. A better way is to only use network API once to have all entities identified: serialize the whole document, indexing each word in the document and being able to recover the original structure after the API returns serialized results. 10x speed for the code.

5. Leave a lot of test suits: to keep tract of how many questions and special types have been solved. When new codes are introduced to solve a new problem, make sure that it doesn’t break previous solved problems.

Cognitive:

## Explain Naive Bayes Algorithm in classifying spam and non-spam

* Problem formulation: P := P(y|(x1, x2,…,xn))
* Bayes rule: P = P(y) \* P(x1, x2,…,xn)|y) / P(x1, x2,…,xn)
* Remove the evidence (denominator) since it doesn’t depend on y: P ~ P(y) \* P(x1,x2,…,xn)
* Joint probability: P ~ P(x1,x2,…,xn, y)
* Joint probability to conditional probability:

P ~ P(x1|x2,..,xn,y)\*P(x2,..,xn,y) = P(x1|x2,..,xn,y) \* P(x2|x3,..,xn,y) = … =

* Naive Bayes assumption: P ~ previous equation = , therefore NB is a MAP classifier