**Extract Data from Resumes**

**Hyperscience → Student Access**

* Document - identification classify
  + Detect if it is a resume
* Extract
  + Structured
    - Not working with these documents
  + Unstructured
    - EX: Legal documents (not really working with these)
  + Semistructured
    - EX: Invoice or like a check (Yes working with it)
  + Deploy
    - Process a minimum of 500,000 - per year
    - Web application that is easily to deploy
    - 200 examples → train a model
    - Extraction is pre-trained
    - Comperical starting price is $150,000

**Google NLP**

* AutoML Text & Document Classification
  + AutoML Sentiment Analysis
    - Single label classification
      * classifies documents by assigning a label to them
      * Sentence is labeled as a specific characteristic
        + EX: Went on a date with Kunal → affection
    - Multiple label classification
      * allows a document to be assigned multiple labels
      * I think overall document classification
    - Each Label must reach 10 documents
    - Consider using “none of the above” characteristic
  + AutoML Entity Extraction
    - identifies entities in documents
    - JSONL format for data
    - In each sentence, there are different words that specify each entity
      * EX: Went on a date with Kunal → Kunal → name (start, end char)
    - [Entity Extraction](https://cloud.google.com/natural-language/automl/docs/prepare?_ga=2.194006317.-1669251217.1592977096#expandable-2)
  + Sentiment Analysis
    - analyzes attitudes within documents
    - Understand the attitude with each sentence with a number
    - Running in PY
      * [Sentiment Analysis Tutorial | Cloud Natural Language API](https://cloud.google.com/natural-language/docs/sentiment-tutorial)
  + [Quickstart | AutoML Natural Language Documentation](https://cloud.google.com/natural-language/automl/docs/quickstart)
  + [How-to Guides | AutoML Natural Language Documentation](https://cloud.google.com/natural-language/automl/docs/how-to)
  + My opinion
    - I really like the software that Google offers, and it is very easy to follow the How-to Guides.
    - Entity extraction would be the best one to use - get specific information from every document (only gets words)
    - Use multiple label classification to extract sentences from full document that contain important information
      * Use automatic sentence splitting and
    - Sentiment Analysis is attitude → not useful
* Cloud Natural Language API

**Document AI**

* Entity Extraction
  + Known entities in documents → proper nouns such as
    - Public figures, company branding
* Parse using tables
  + Don’t completely understand, call to get to know more
* <https://cloud.google.com/document-ai/docs>

**H2O AI**

* 21 day trail
  + Open source verision
* Word2Vec algorithm → NLP with H2O
  + Detecting whether a series of sentences or words are positive or negative
  + Detect between 2 things
  + Could use between
* Builds a Model that trains for sentiment analysis → 2 main things
  + extract opinions from a given text → attitudes, sentiments, and emotions
  + Rate these attitudes numerically → 1-5
* High quality model with high accuracy and speed and run on web interface
  + [Using Flow - H2O's Web UI](https://docs.h2o.ai/h2o/latest-stable/h2o-docs/flow.html)
* [Github Tutorial for Webflow with Sentiment Analysis](https://github.com/h2oai/tutorials/blob/master/Driverless%20AI/natural-language-processing-tutorial-sentiment-analysis/natural-language-processing-tutorial-sentiment-analysis.md#task-1-launch-sentiment-analysis-experiment)
* [Creating a Binary Classifier to Sort Trump vs. Clinton Tweets Using NLP](https://medium.com/the-story-within/creating-a-binary-classifier-to-sort-trump-vs-clinton-tweets-using-nlp-3e19e8183409)
* [NLP with H2O | H2O Tutorials](https://docs.h2o.ai/h2o-tutorials/latest-stable/h2o-world-2017/nlp/index.html)
* [Natural Language Processing in H2O's Driverless AI - Open Source Leader in AI and ML](https://www.h2o.ai/blog/natural-language-processing-in-h2os-driverless-ai/)
* Things it can do
  + Document Classification → whether overall document is what
  + Sentiment Classification → text is positive, negative, or neutral - number
  + Text and numerical data → understand number and text hybrid
* DriverlessAI
  + Count based words
  + TF-IDF → Most important words
  + Word Embeddings → semantics captured by representing words in higher dimensions
  + Stemming and Lemmatization
  + Part of speech tagging
  + Spelling correction
* May allow Custom models to the recipe with NLP H2O pipeline

**Things to extract from the resume**

* Name
* Email
* LinkedIn
* Address
* Phone Number
* List of skills proficient in → Skills
  + Technologies; Programming; ETC
* Certifications/Awards won
* Qualifications
* Work Experience
* Career objectives
* Education

**Things to look out for**

* Work with full document → Location of document affects

**Lexical Analysis**

* A type of compiler
* Converts code into tokens to be then sent into later steps to be processed
* First removes comments, then takes text and extract each character → Lexemes
* Then use a tokenizer to gets main words and then run through compiler

**NLP Details**

* Sentences
  + Split into tokens
  + Using the defining linguistic features to detect specific words
  + Get needed words from dataset and using spacy scan each token
  + Scan if the defining features match
    - Text: The original word text.
    - Lemma: The base form of the word.
    - POS: The simple UPOS part-of-speech tag.
    - Tag: The detailed part-of-speech tag.
    - Dep: Syntactic dependency, i.e. the relation between tokens.
    - Shape: The word shape – capitalization, punctuation, digits.
    - is alpha: Is the token an alpha character?
    - is stop: Is the token part of a stop list, i.e. the most common words of the language?
  + Detect if skill is detect and if correct
  + Problems faced
    - Misses a lot of the things
    - Write code to create a benchmark to check if it works and understand the many different occasions this can happen and re add it into the code
  + Start editing own model
    - Use different previous examples as data
* Bag of words algorithm → work

<https://medium.com/searce/tips-tricks-for-using-google-vision-api-for-text-detection-2d6d1e0c6361>

* Sentence List → OVERALL LIST
  + Sentence 1
    - Text → “NISARGA HASSAN SREEDHAR”
    - Attribute 1 → bracket
      * Attribute Name → “NAME”
      * Word → “NISARGA HASSAN SREEDHAR”
      * Start Char → 0
      * End Char → 25
  + Sentence 2 → “Programming: Python, Java”
    - Attribute 1 → bracket
      * Attribute Name → “Programming language”
      * Word → “Python”
      * Start Char → 10
      * End Char → 14
    - Attribute 2 → bracket
      * Attribute Name → “Programming language”
      * Word→ “Java”
      * Start Char → 15
      * End Char → 21