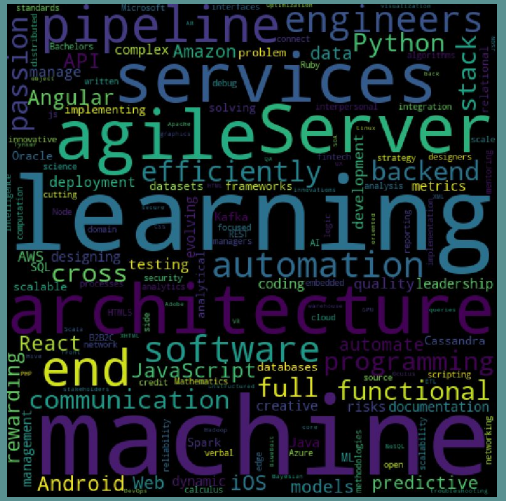
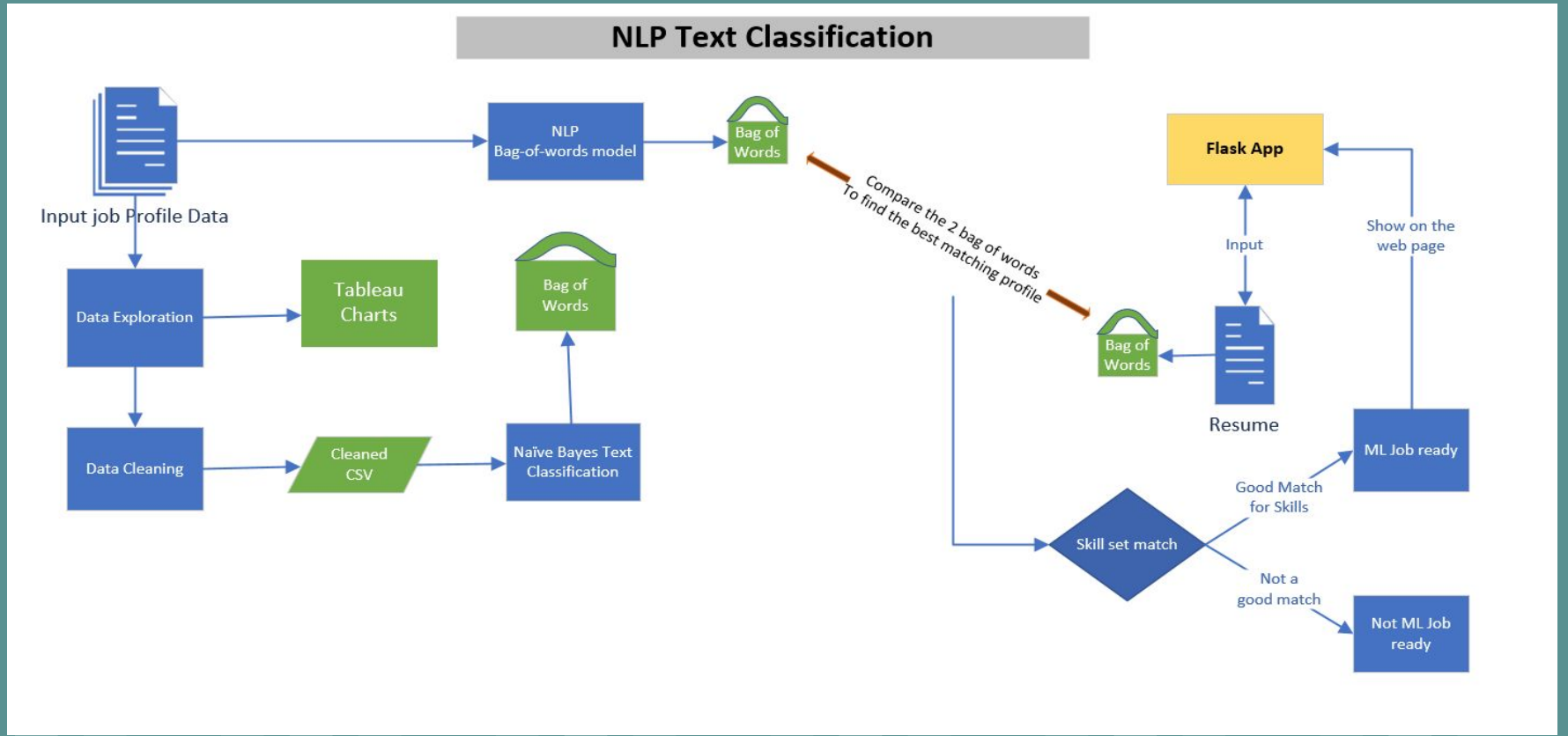
Skills and Jobs Match using NLP

Contributors: Bhumika Sharma
Cong Tran
Denis Cohen
Chris Janssen
Imran Bawany
Vijay Gangaprasad



Goal of Project

1. Help Job seekers to find how their skills stack up against the most sought after Machine Learning skills and tools and also recommend job postings that matches their skills reflected in their resume.
2. We focused on Machine Learning type job postings as the basis of the application but the functionality can extend to all types of postings.
3. We used NLP techniques to build our matching process with machine learning libraries in Python.


NLP Text Classification
Input job Profile Data
NLP
Bag-of-words model
Bag of
Words
Compare the 2 bag of words
To find the best matching profile
Flask App
Input
Show on the
web page
Data Exploration
Tableau
Charts
Bag of
Words
Bag of
Words
Resume
Data Cleaning
Cleaned
CSV
Naïve Bayes Text
Classification
ML Job ready
Good Match
for Skills
Skill set match
Not a
good match
Not ML Job
ready

Data

Source

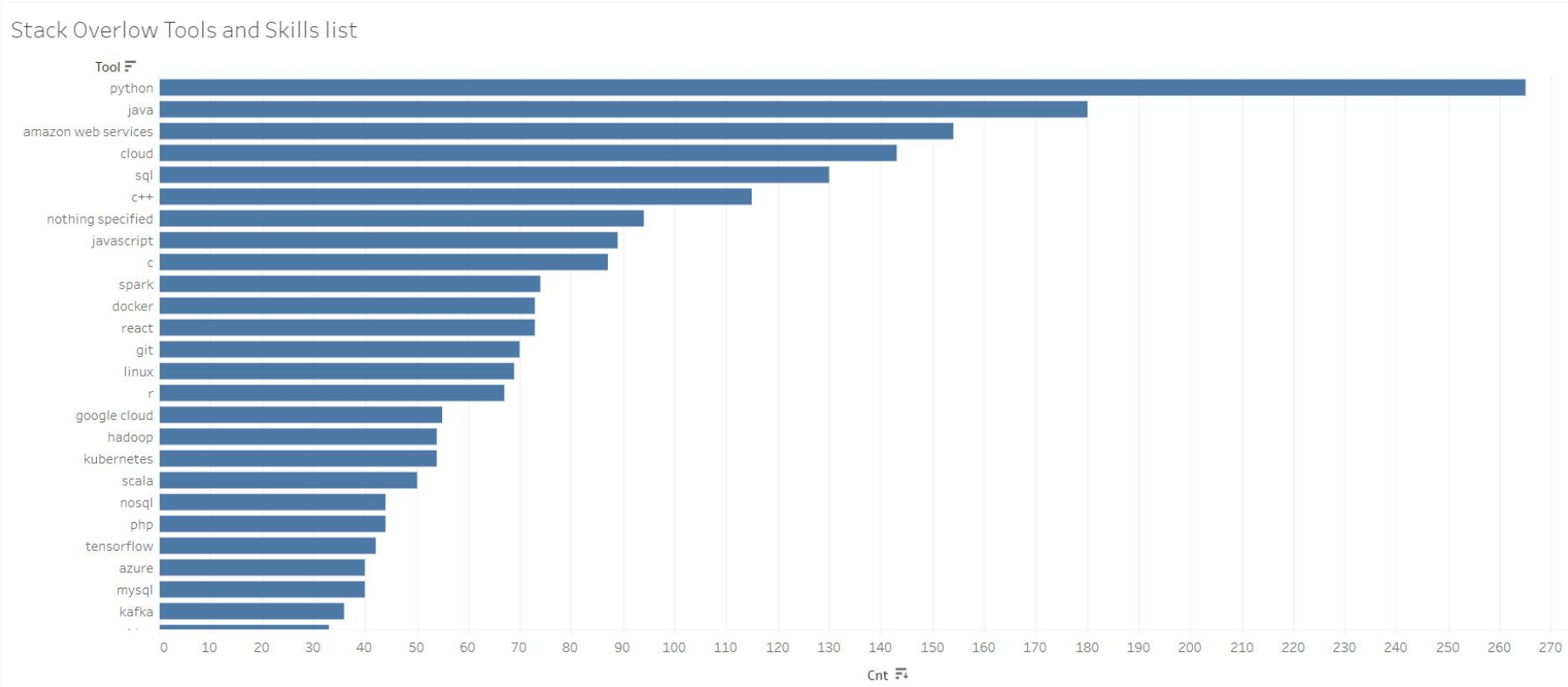
Retrieved data set of job postings from Stack Overflow using the RSS Feed downloaded as an XML file and converted to a CSV.

Exploration and Cleansing

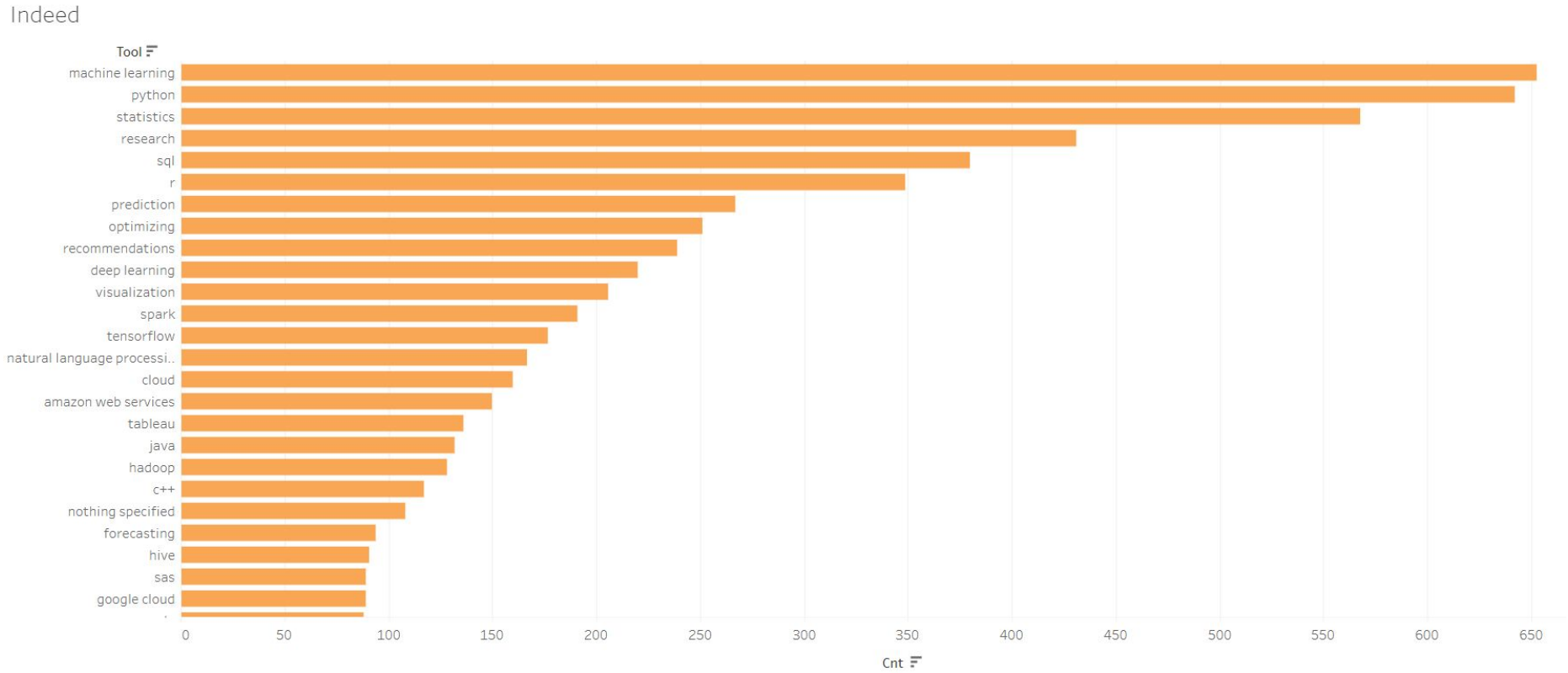
Used Tableau to build chart to visualize the data. Visualizations included Job Title by Company, Job count by State, and Job count by Company. Data cleaning is done using Python libraries like Pandas.

Issues

The biggest issue was finding a dataset of current job postings. Many of the job posting websites (Indeed, Monster, etc) did not have an accessible API.

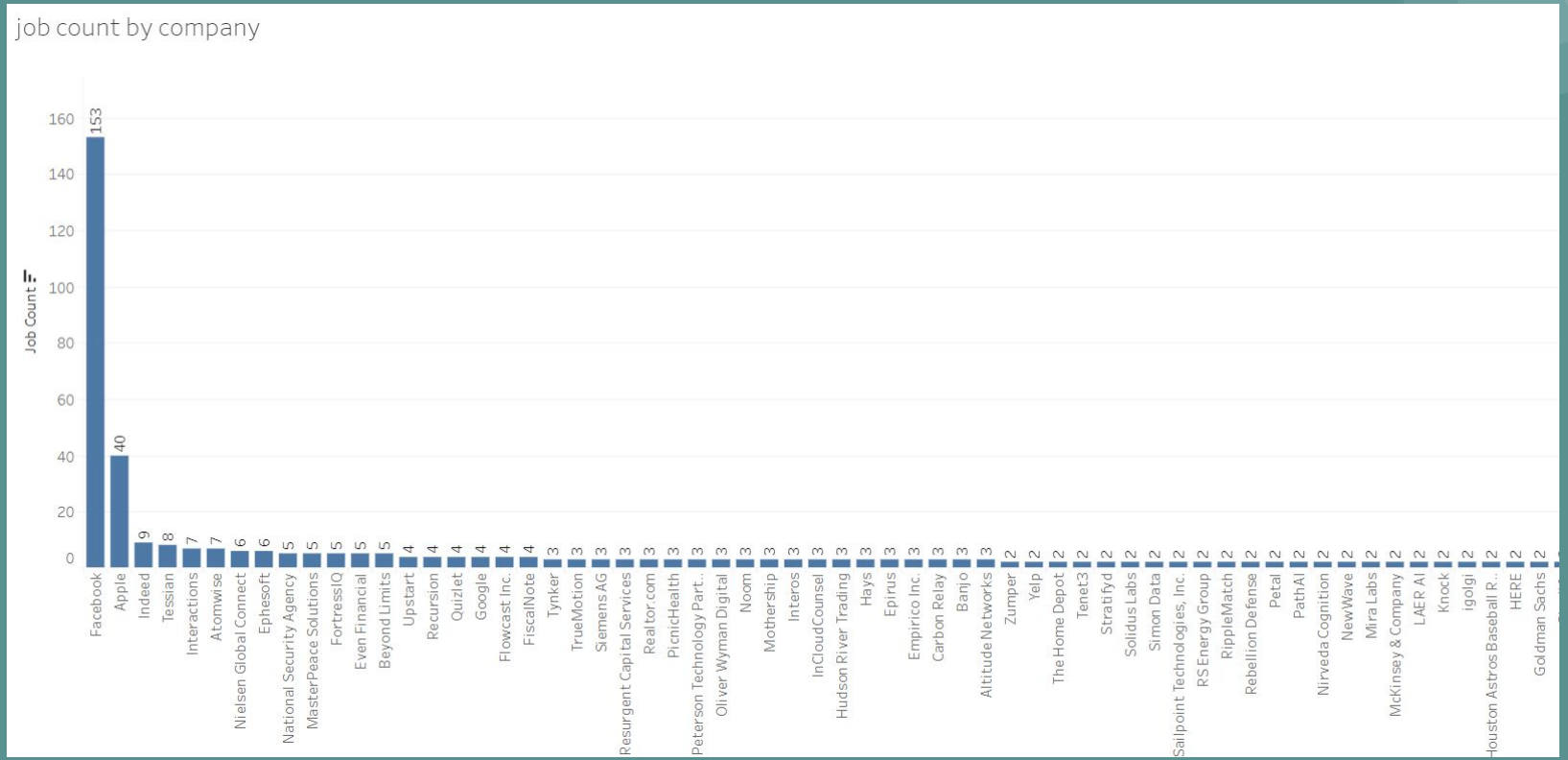


https://public.tableau.com/profile/vijaybabu.gangaprasad#!/vizhome/StackOverflowToolsandSkills/Sheet1

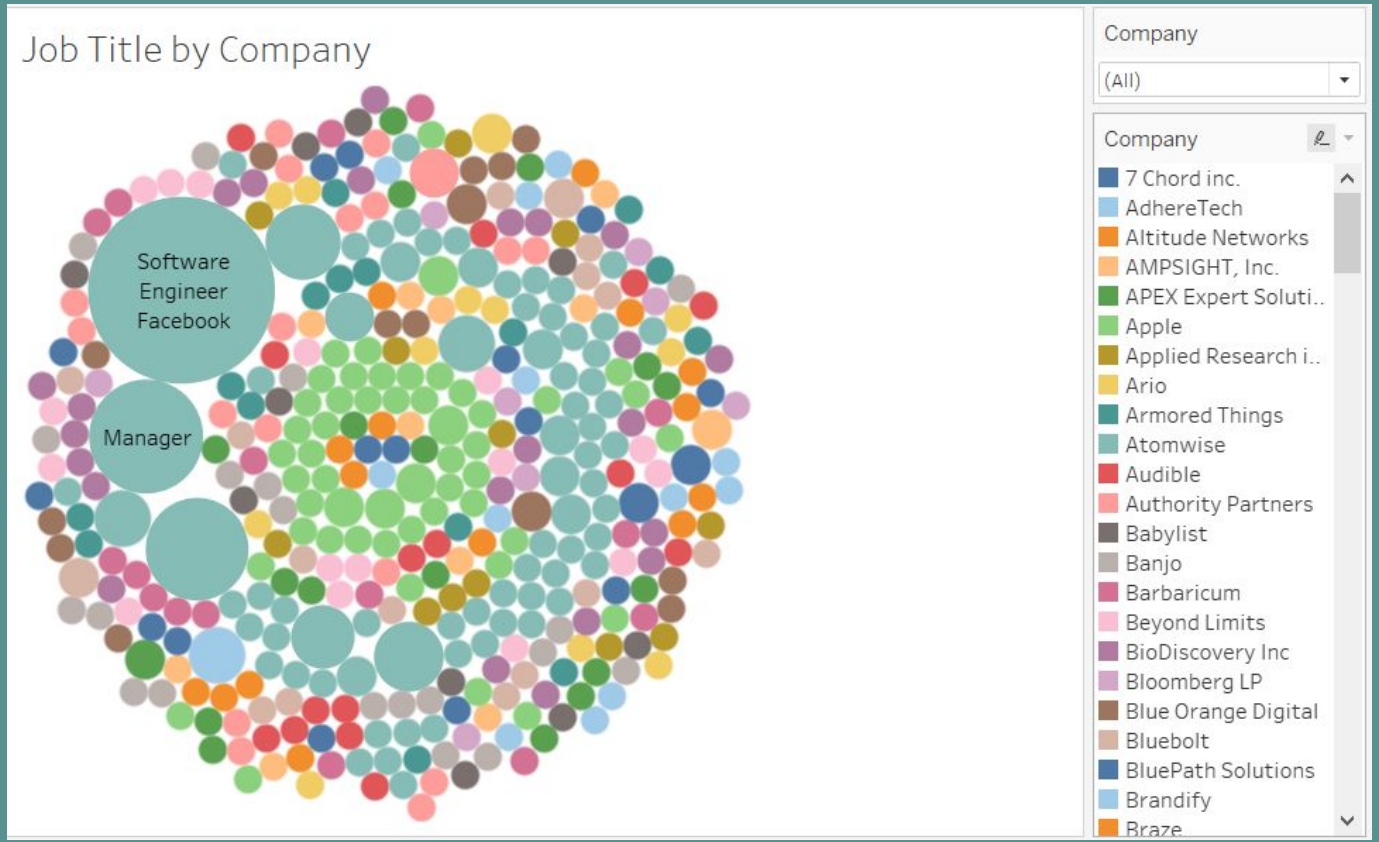


https://public.tableau.com/profile/vijaybabu.gangaprasad#!/vizhome/IndeedSkillsandtools/Indeed

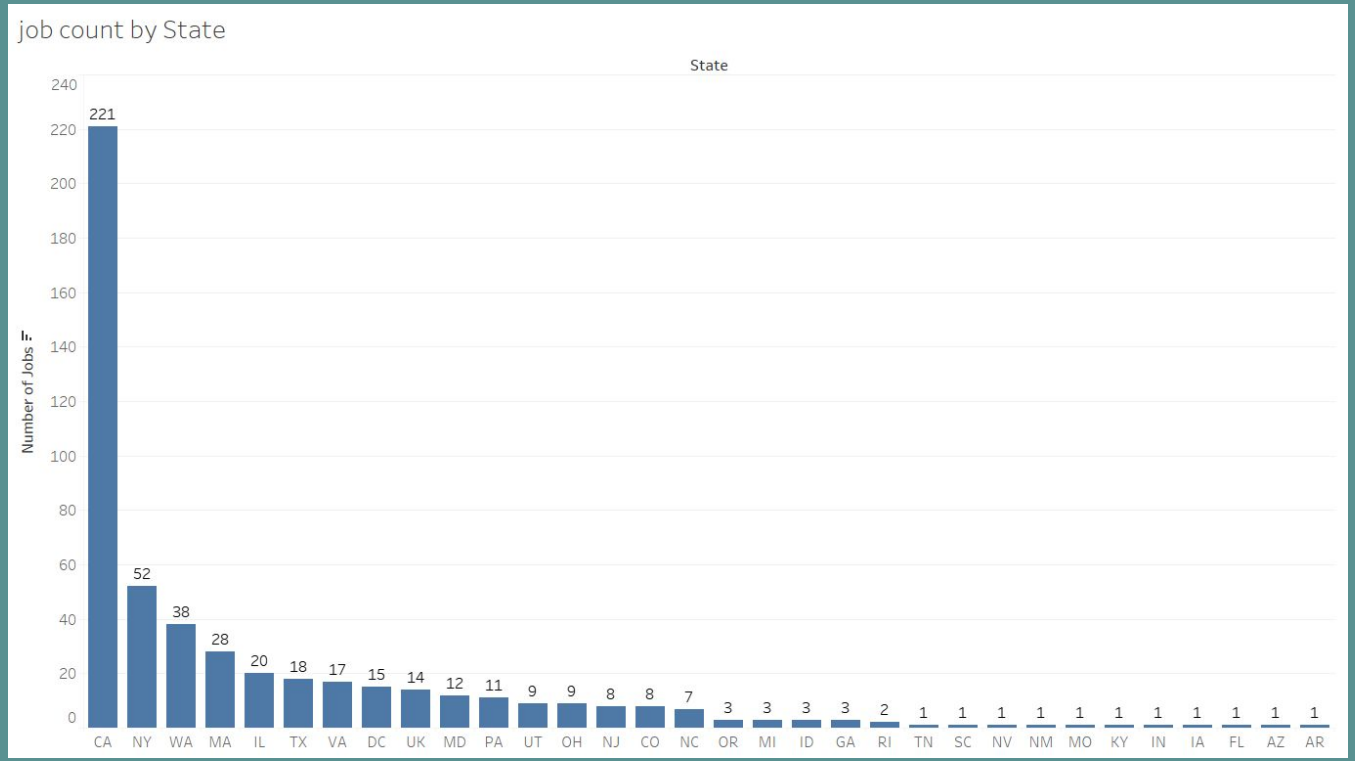
Visualizations: Job Count by Company



Visualizations: Job Title by Company



Visualizations: Job Count by State



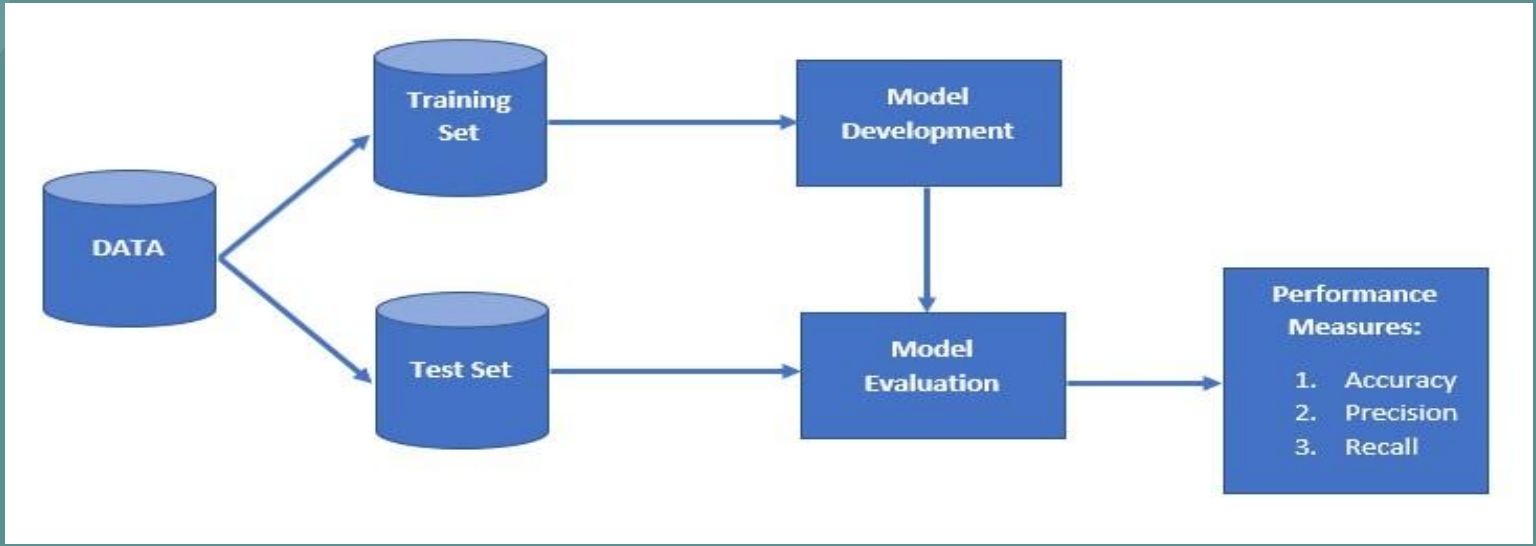
Naive Bayes Classifier for Text Classification

Naive Bayes classifiers, a family of classifiers that are based on the popular Bayes' probability theorem, are known for creating simple yet well performing models, especially in the fields of document classification and disease prediction.Naive Bayes classifiers are linear classifiers that are known for being simple yet very efficient.
In our case,the algorithm is applied to classify the job listing as either 'Machine Learning' or not.

Steps:

- Data collection :StackOverflow job listings,data exploration, cleaning
- Create all the features to the data set:Labeling,Tokenization,Stemming,stopword removal,Hashing TF,IDF
- Naive Bayes model and fit training data
- Transform the model with the testing data
- Evaluation
- Prediction

Naive Bayes Classification Workflow

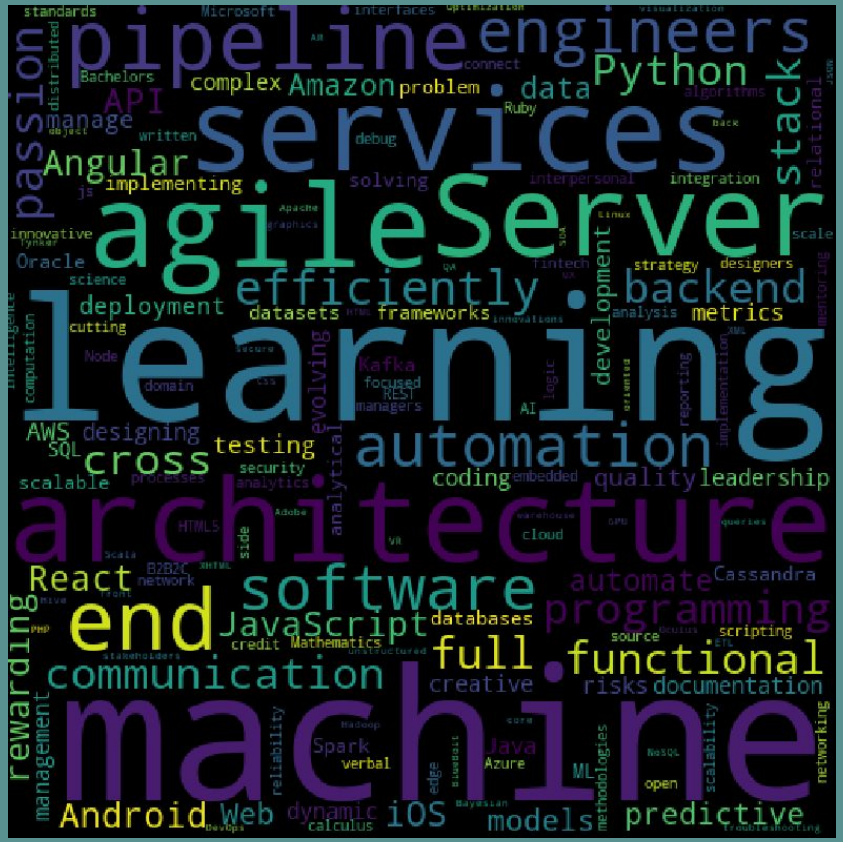


Terms:

*Tokenization:*general process of breaking down a text corpus into individual elements
*Stemming:*process of transforming a word into its root form
Stop Words:words that are particularly common in a text but rather un-informative (e.g., words such as so, and, or, the, …")
*Bag of Words:*a collection of words from a text with word count and disregarding the order in which they appear
TF(Term Frequency):number of times a word appears in a document divided by total number of words in the document
*IDF(inverse document frequency):*measures how important a term is.
IDF(t) = log_e(Total number of documents / Number of documents with term t in it)
Data Pipeline:data processing elements connected in series, where the output of one element is the input of the next one

Word Cloud: Words Before Cleanup

Word Cloud: Machine Learning Focus



Process for App

Design --->

Used NLP techniques to design application using Python to match the keywords from user input resume to match with the keywords in the corpus of job postings. The application was based on the ranking results of TF-IDF.

Flask App --->

Built Flask App to serve our application and created a user interface with HTML to allow the user to enter their resume as text. The application is then able to run the resume through the NLP model to create the match.

Hosting

Used AWS to host the server and datasets in S3 buckets. We are able to launch the application in the cloud through AWS.

Programs/Libraries/Languages/Services Used

- Python
- Flask App
- HTML
- Pandas
- NLTK
- Gensim
- AWS
- Tableau
- Scikit-Learn
- Matplotlib
- WordCloud
- Numpy
- Math

Demo of Resume and Job Posting Application

Github

Please visit our Github repository to see code and other resources:

https://github.com/vgangaprasad/ML_Skills_Match

Questions?