

TEXT ANALYTICS PROJECT

User Manual

Team Members: Qianwen Chen, Shruti Bakare, Aditee Verma, Christopher Casasanta, Christian White

Customers: Andreea Cotoranu and Avery Leider

Seidenberg School of CSIS, Pace University, New York

Table of Contents

1. Introduction..........................................................................................1

2. Acronyms and Abbreviations………………………………………...2

3. System Overview……………………………………………………..3

4. Getting Twitter APIs………………………………………………….4

5. How to install Anaconda on Windows 10……………………………5

1. INTRODUCTION

This study performed text analytics on a large dataset to create a thesaurus of cybersecurity terms. The dataset consisted of 70,115 tweets from 25 expert users. By selecting the top experts in the field and analyzing their tweet texts, the study built a thesaurus of expert related cybersecurity terms. Two text processing tools were tested for this task by running the twitter data through Python and RapidMiner and then analyzing the results to choose the best method. To further the analysis and produce the best results, the profiles were split into two groups. Each group was then processed separately by applying the TF-IDF method and observing the similarities between the results. This enabled the study to derive an accurate dictionary of cybersecurity terms.

The purpose of this work is to create a thesaurus from tweets about cybersecurity for a group of experts. The primary steps to create the thesauri are:

1) Identify 25 frequent tweeters on cybersecurity that can be classified as “experts” in the field of cybersecurity.

2) Create a data set of all tweets of each tweeter and compile all tweets into CSV files.

3)Run text analysis on each user’s data set using top two preprocessing methods:

a. Python Module

b. RapidMiner

4) Split data set from Python into two groups.

5) Run frequency analysis on each group using TF-IDF method.

6) Compare TF-IDF results to discover text similarities among each expert.

1. Acronyms and Abbreviations

NLTK: - Natural Language Toolkit

PoS Tagging: - Part of Speech Tagging

TF: - Term Frequency

IDF: - Inverse Document Frequency

NLP: - Natural Language Processing

1. System Overview

In this project, we have collected tweets from 25 expert Twitter users related to Cybersecurity field. And we have separated these 25 expert users in 2 groups: Group A and Group B.

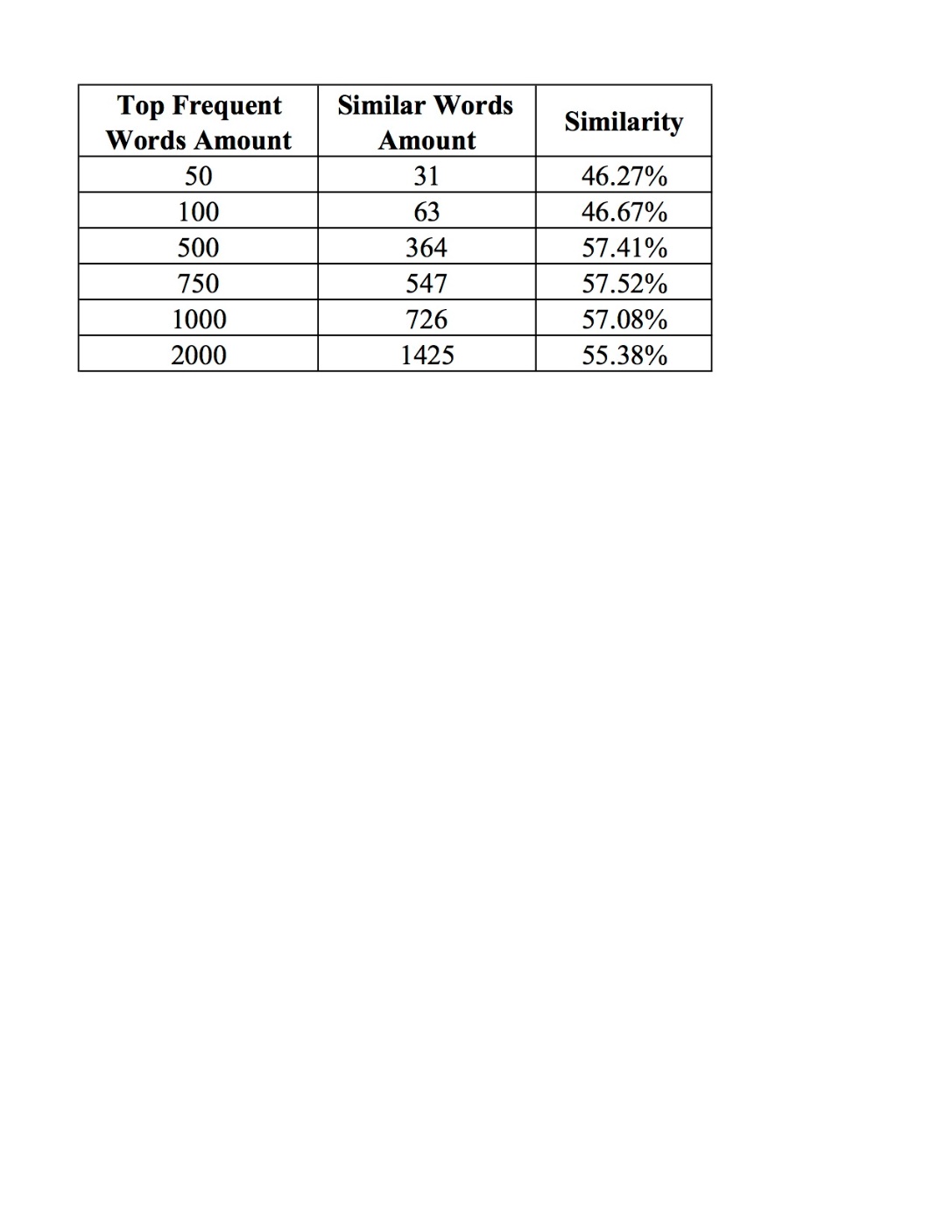
Total Number of tweets collected: 70115

|  |  |
| --- | --- |
| **GROUP A - 37,807 Tweets** | **GROUP B - 32,308 Tweets** |
| @Cyber | @CLTCBerkeley |
| @NJCybersecurity | @NISTcyber |
| @NortonOnline | @DarkReading |
| @TheCyberSecHub | @threatintel |
| @symantec | @McAfee |
| @CheckPointSW | @PaloAltoNtwks |
| @TrendMicro | @Fortinet |
| @Stanford\_Cyber | @proofpoint |
| @Incapsula\_com | @CyberArk |
| @avgaunz | @Gemalto |
| @LifeLock | @ManTech |
| @splunk | @VERISIGN |
| @digicert |  |

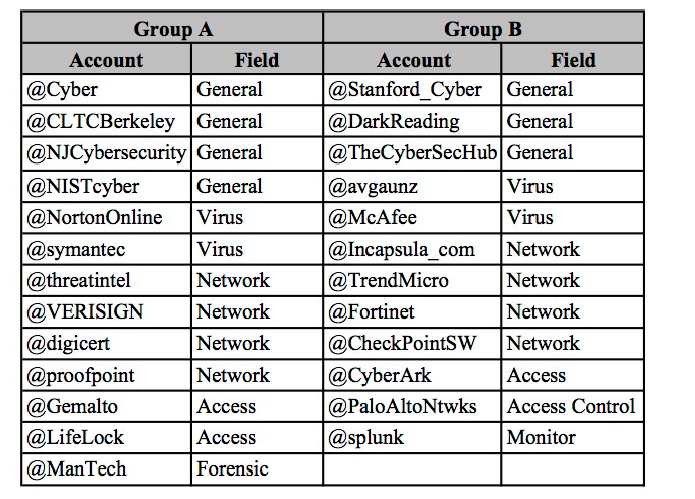
The python modules and other related files are hosted over github at this link: <https://github.com/averma74/Text-Analytics>

The project involves the below python modules:

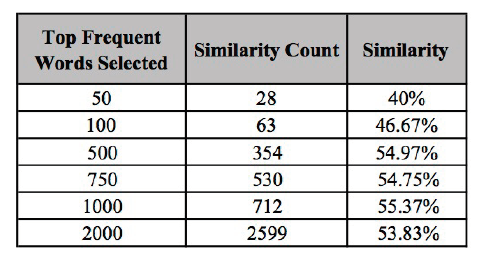
1. **tweet\_dumper.py**: It's the tweepy python program file. In this python file, we will get our twitter data using tweepy library. Initially, we need the Twitter API credentials so that we can access the twitter data. Section 4 explains how to extract them. The API credentials are namely as consumer\_key, consumer\_secret, access\_key, access\_secret. We just have to put the twitter id in the program and it gives a csv file of the id's tweeter data. This twitter data contains 3 columns namely id, date and time of the tweet and the actual tweet.
2. **MainProgram.py:** This is the program which performs the data preprocessing. We are providing the output of tweet\_dumper.py to this code. In this python file, all the preprocessing tasks like cleaning the data (Removing Unicode, numbers, tweeter username, stop words), tokenizing, PoS tagging and lemmatization is done. Initially, we are reading our twitter data from csv file. Then we are making each tweet lower case. The next step is to remove all the unicodes, links, tweeter username and stop words. The next step is to tokenize the tweets so that we can easily separate the words in the sentences. After tokenization, Part of Speech tagging is done, which means we are classifying words according to the noun, verb, adjective, etc. Lemmatize function is used to find the lemma or root of the word. Again PoS tagging is performed since it is removed in lemmatization process. After this, we are going to check whether the given word is noun and store that word in the frequency list. If the given word is not present in the frequency list, then add that word in the frequency list. If the given word is present in the frequency list, increase the frequency counter. After this, we are going to sort the dictionary according to the frequency and store the result in csv file where frequency is written in front of each word. The output is the final thesaurus for the given profile.
3. **Tfidf.py**: This is the program which is used to calculate Term Frequency and Inverse Document Frequency. We combined all the group A user’s thesaurus into one csv file and similarly combined for Group B users and use these two files as the input to this program. In this program, we are reading the two columns from the csv file which contains words and its frequency. The words from each csv files are stored in separate bag of words. Using union function, we can join these two strings from 2 different csv files. Then we are creating a dictionary to keep the word count for each user. The words are counted from the bag of words for each user. After this, we have to create matrix of words. Using computeTF() function we can rank the words according to the relevance. After this we are going to compute inverse document frequency. To calculate IDF, first count the number of tweets that contains particular word, divide it by total number of tweets and take log of that division. In this way, we will get TF and IDF. We are storing this result in the form of csv file. This csv file contains 3 columns namely word, TF-IDF of group A and TF-IDF of Group B.
4. **Similarity.py**: This program is used to find the similar words between the expert users. The input to this python file is output of tfidf.py file. If the value of tf-idf is 0 that means the word is present in the respective group, hence is for both group if tf-idf value is 0, it shows that the word is present in both the groups. We check this in the document and then calculate the percentage of similar words for both the groups. We ran this program by taking different range of words like top 50, 100, 500, 750, 1000 and 2000. Below are our results:



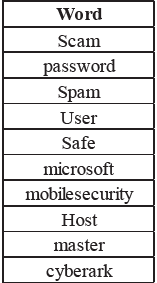
To get better results we tried to create groups based on the cybersecurity expert’s field and created the groups again as below:

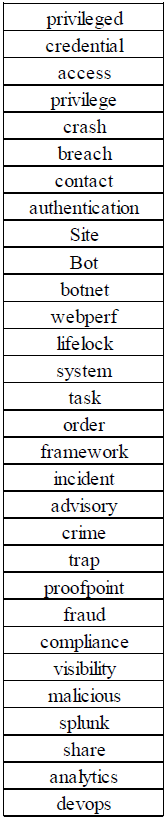


We did the similar processing for these groups and found below results:



1. **tfidfDocuments.py**: We took the top 20 frequent words of each of the 25 cybersecurity experts, making the total word count 500 words. We used this module to calculate the TF-IDF ‘score’ for these 25 documents. The higher the ‘score’, then the more important the word. After running the 25 documents through this Python program, the highest TF-IDF score came out to “0.12629”. 40 total words with the high score are listed below:





1. Getting Twitter API keys

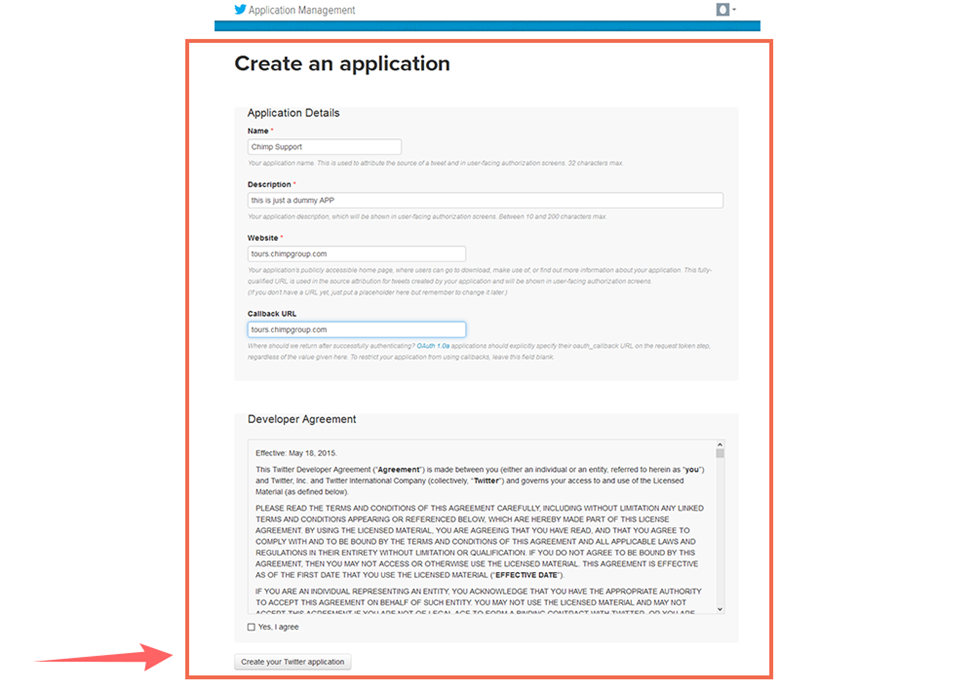
To get Twitter Access keys, you need to create Twitter Application which is mandatory to access Twitter. To start with, you will need to have a Twitter account and obtain credentials (i.e. API key, API secret, Access token and Access token secret) on the Twitter developer site to access the Twitter API, following these steps:

1. Create a Twitter user account if you do not already have one.

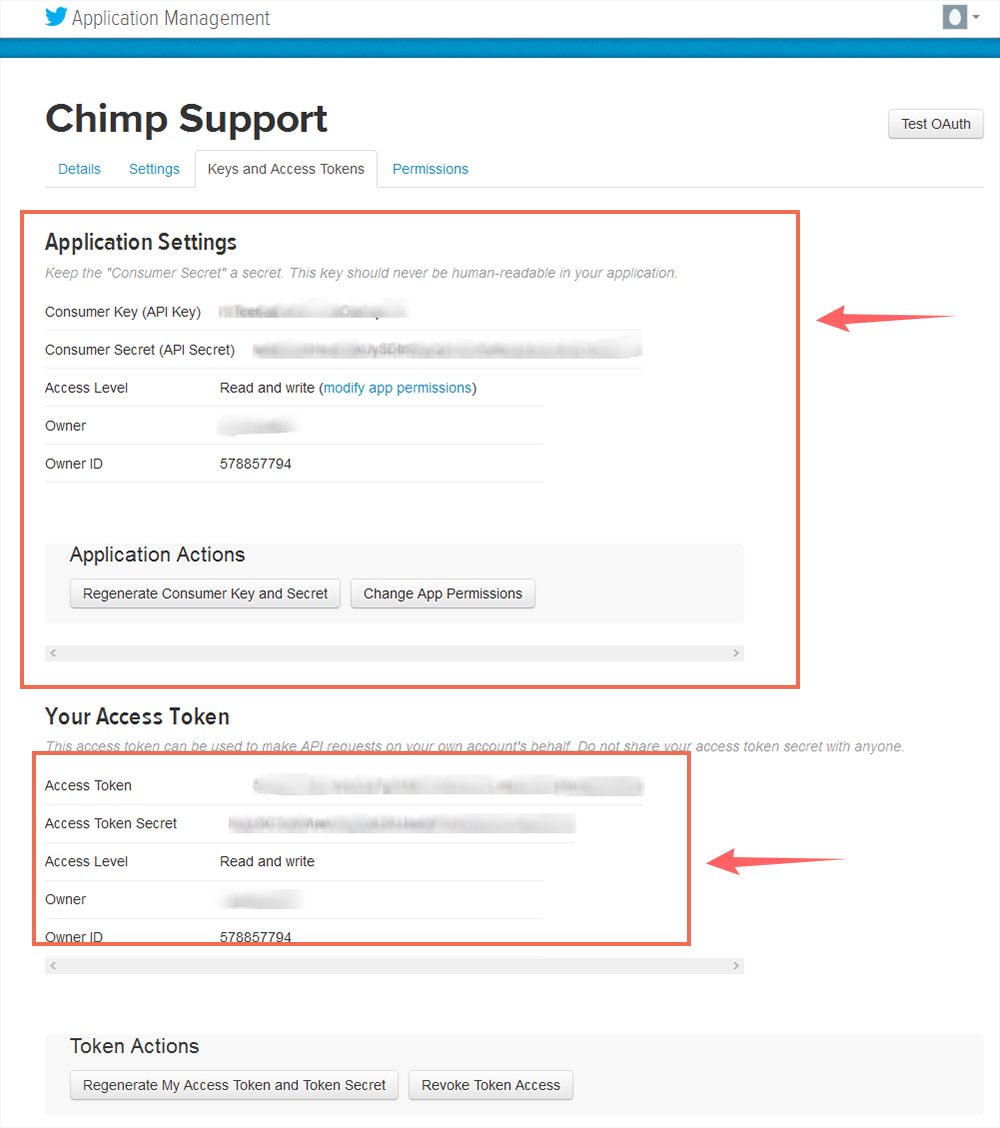
2. Go to https://apps.twitter.com/ and log in with your Twitter user account. This step gives you a Twitter dev account under the same name as your user account.

3. Click “Create New App”

4. Fill out the form, agree to the terms, and click “Create your Twitter application”



5. In the next page, click on “Keys and Access Tokens” tab, and copy your “API key” and “API secret”. Scroll down and click “Create my access token”, and copy your “Access token” and “Access token secret”.



5. HOW TO INSTALL ANACONDA IN WINDOWS 10?

Since we have used Anaconda navigator to run our Python program, here are the steps to install Anaconda on Windows machine:

**What is ANACONDA?**



Anaconda Distribution is the most popular python Data Science platform. It is a free, easy- to-install package manager, environment manager and Python distribution with a collection of 1,000+ open source packages with free community support.

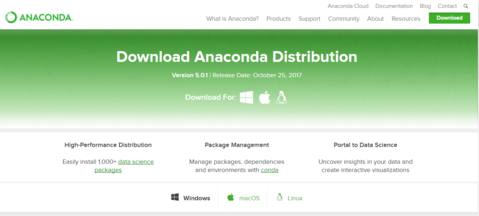
**How to install?**

It is a 10-step easy installation guide for Anaconda Distribution for Windows 10 users.

**Step 1:**

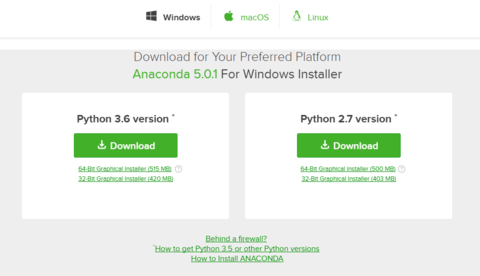
Visit the download page for Continuum Analytics to download the free “Anaconda” Python distribution.

Download link: <https://www.anaconda.com/download/>



**Step 2:**

Select the "Operating System" (Here Windows 10) and Download Anaconda installer for preferred "platform" (python 3.6 or 2.7).



**Step 3:**

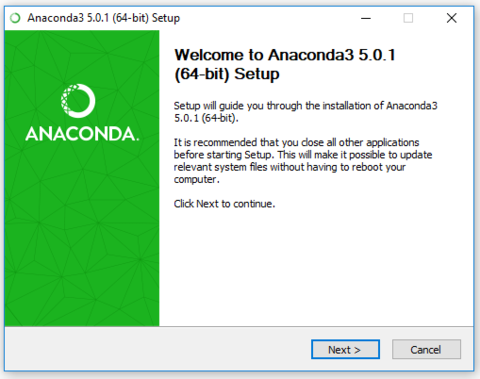
Go to the download location and “Double click" the installer to launch.

https://cdn.shopify.com/s/files/1/2419/2501/files/installer_large.PNG?v=1510042523

NOTE: If you encounter any issues during installation, temporarily disable your anti-virus software during install, then re-enable it after the installation concludes. If you have installed for all users, uninstall Anaconda and re-install it for your user only and try again.

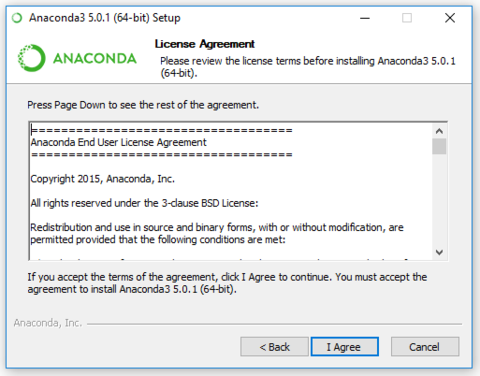
**Step 4:**

Click "Next".



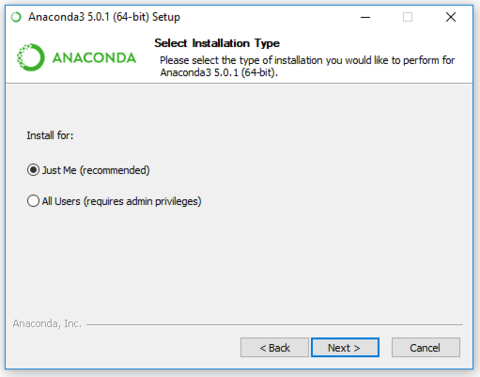
**Step 5:**

Read the licensing terms and click “I Agree”.



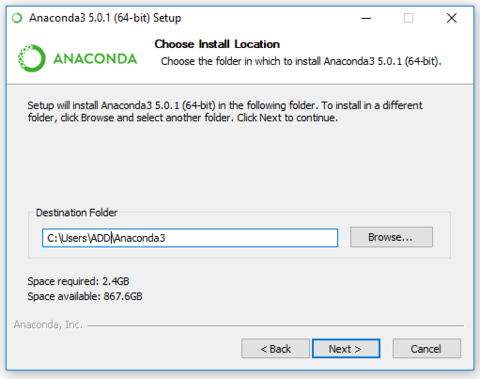
**Step 6:**

Select an install for “Just Me” unless you’re installing for all users (which requires Windows Administrator privileges) and click “Next”.



**Step 7:**

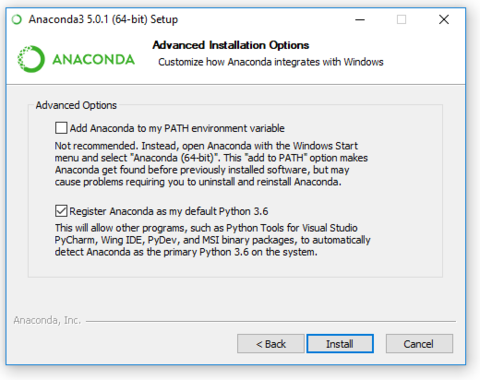
Select a destination folder to install Anaconda and click the "Next" button.



NOTE: Install Anaconda to a directory path that does not contain spaces or Unicode characters. Do not install as Administrator unless admin privileges are required.

**Step 8:**

Choose whether to add Anaconda to your PATH environment variable and Click the "Install" button.

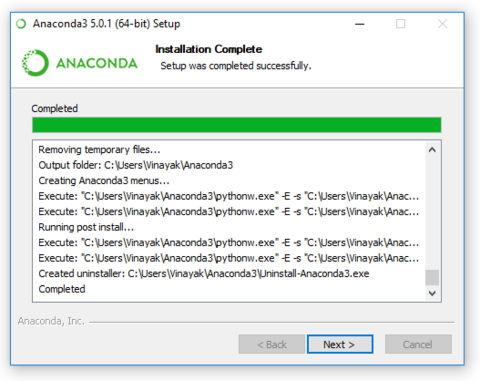


NOTE:  We recommend not adding Anaconda to the PATH environment variable, since this can interfere with other software. Instead, use Anaconda software by opening Anaconda Navigator or the Anaconda Prompt from the Start Menu.

**Step 9:**

Installation Complete! Click the "Next" button.

You can click "Show Details" if you want to see all the packages Anaconda is installing.

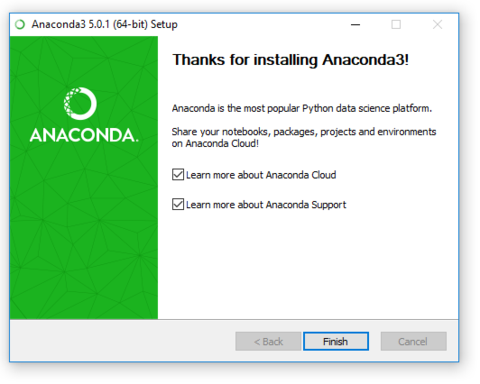


NOTE: Installing may take 15-20 minutes (may be more depending on your internet speed). The installer we’ll be downloading and installing is larger than the average file, because it contains Python and associated packages, editors, and some other tools.

**Step 10:**

After a successful installation you will see the dialog box: “Thanks for installing Anaconda” .

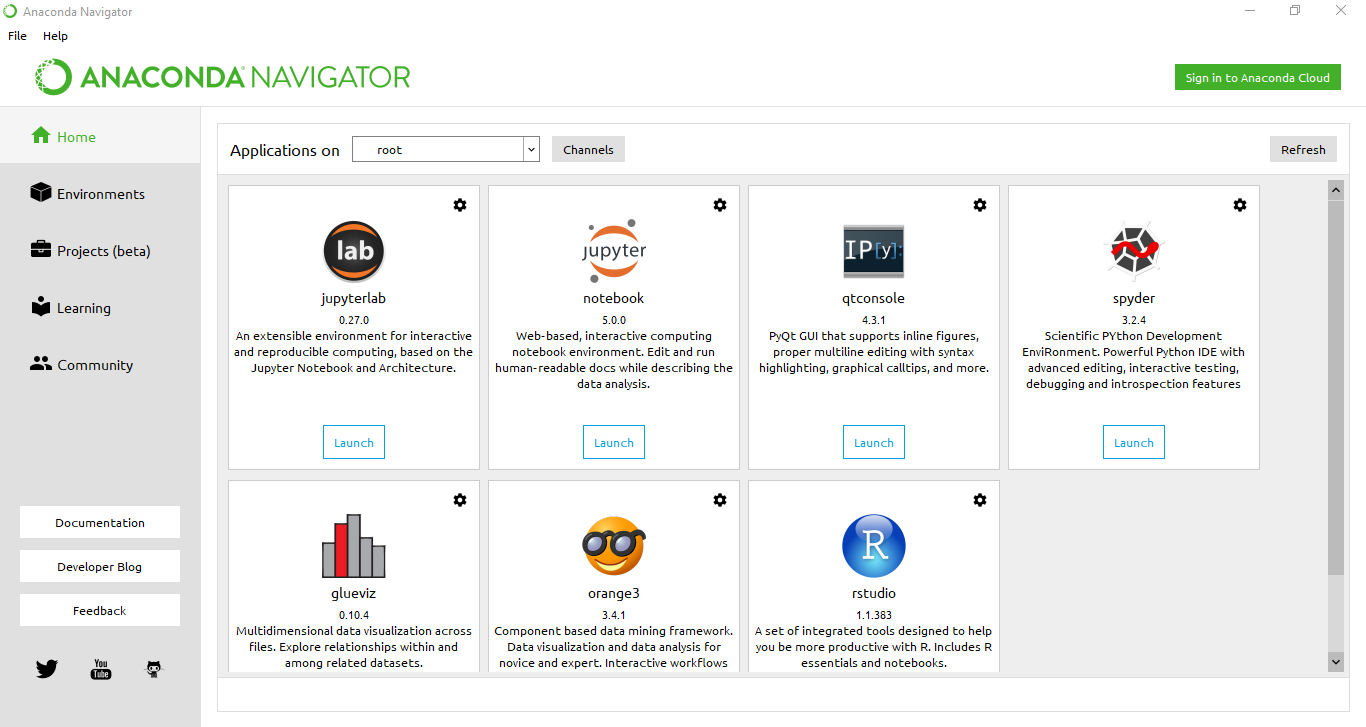
Click "Finish" button.



NOTE: You can leave the boxes checked “Learn more about Anaconda Cloud” and “Learn more about Anaconda Support” if you wish to read more about this cloud package management service and Anaconda support. Click the Finish button.

After your install is complete, verify it by**opening Anaconda Navigator,** a program that is included with Anaconda: from your Windows Start menu, select the shortcut Anaconda Navigator. If Navigator opens, you have successfully installed Anaconda.

Now you can run your Anaconda Navigator.



Launch “spyder” which is an IDE for python and run the python modules through them which are listed over github repository.