Singapore Tourism Board
– Workshop: Survey Analysis

MARCH 3

IBM

Authored by: ANINDYO SARKAR

Use Case Background

Tourist Feedback Analysis

Singapore Tourism Board receives survey / feedback results from tourists for various attractions in Singapore. The intent of this exercise is to analyze the survey results and get meaningful insights from the feedback data that can be used to positively influence the tourist footfall. The feedback is a free form text entered by tourists at different attractions. For our scope we are using sample feedback from Gardens by the Bay. STB has identified the following entities on which feedback needs to be analyzed:

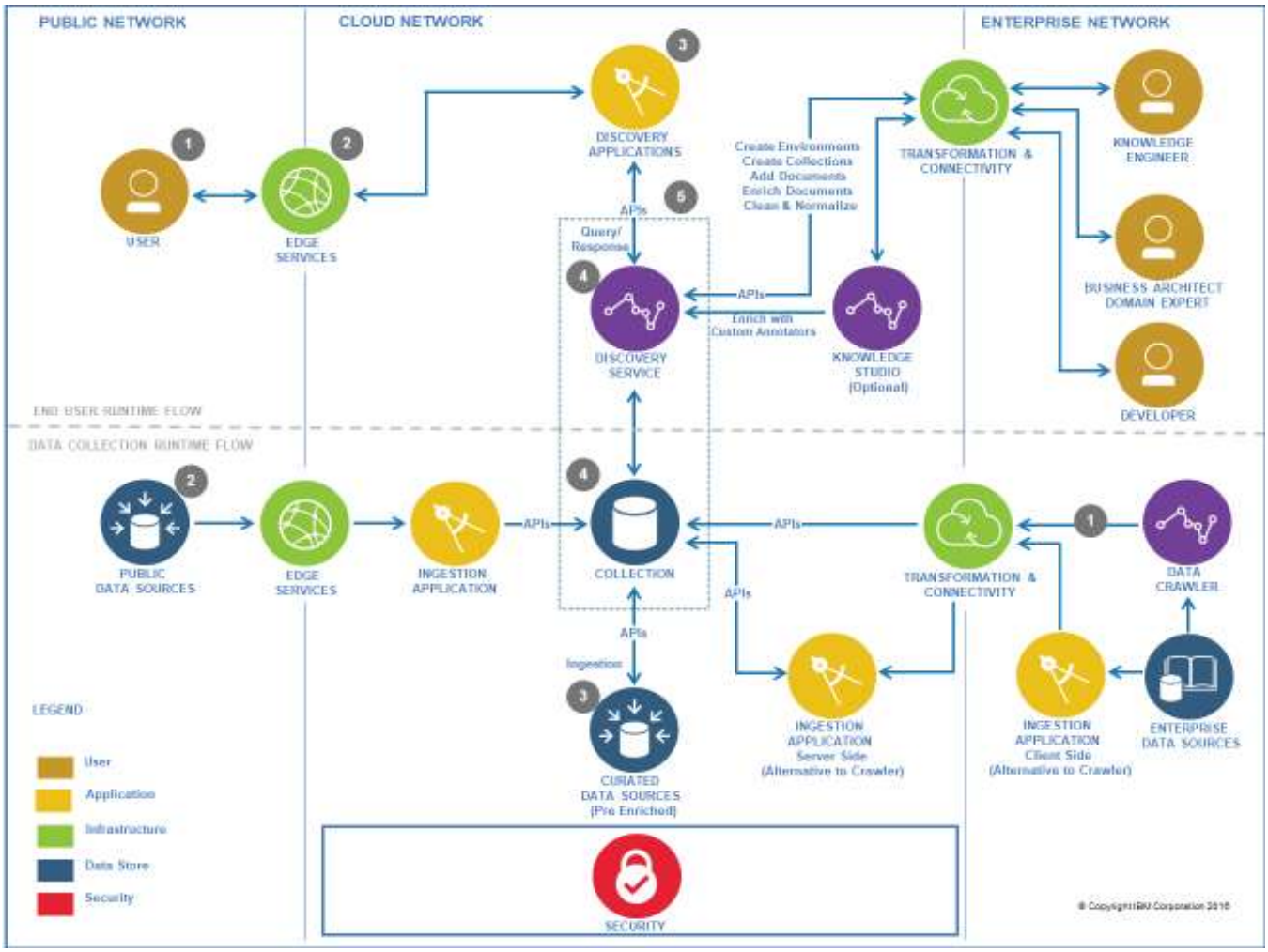
- Sightseeing: View of the place, aesthetics, scenery, ambience etc.
- Transportation: Ease of getting there, transportation details, transport options etc.
- Service Quality: Feedback regarding human services/quality/timeliness etc.
- F&B: Cuisine, culinary or dietary requirements, range, hygiene etc.
- Shopping / Retail: Adequacy, range, variations, relevance etc.
- Information accessibility: Info availability, opening hours, information accessibility
- Facilities: Washroom, aircon, wheelchair accessibility, lifts, escalators, drinking water
- Cultural shows: Events, performances and tours
- Price: Affordability, value of money, ticket prices

The idea of the exercise is to analyze feedback on the above entities and collect useful insights which can be used for future planning.

To perform this exercise, we shall leverage IBM Watson capabilities. IBM Watson has powerful text analytics capabilities such as:

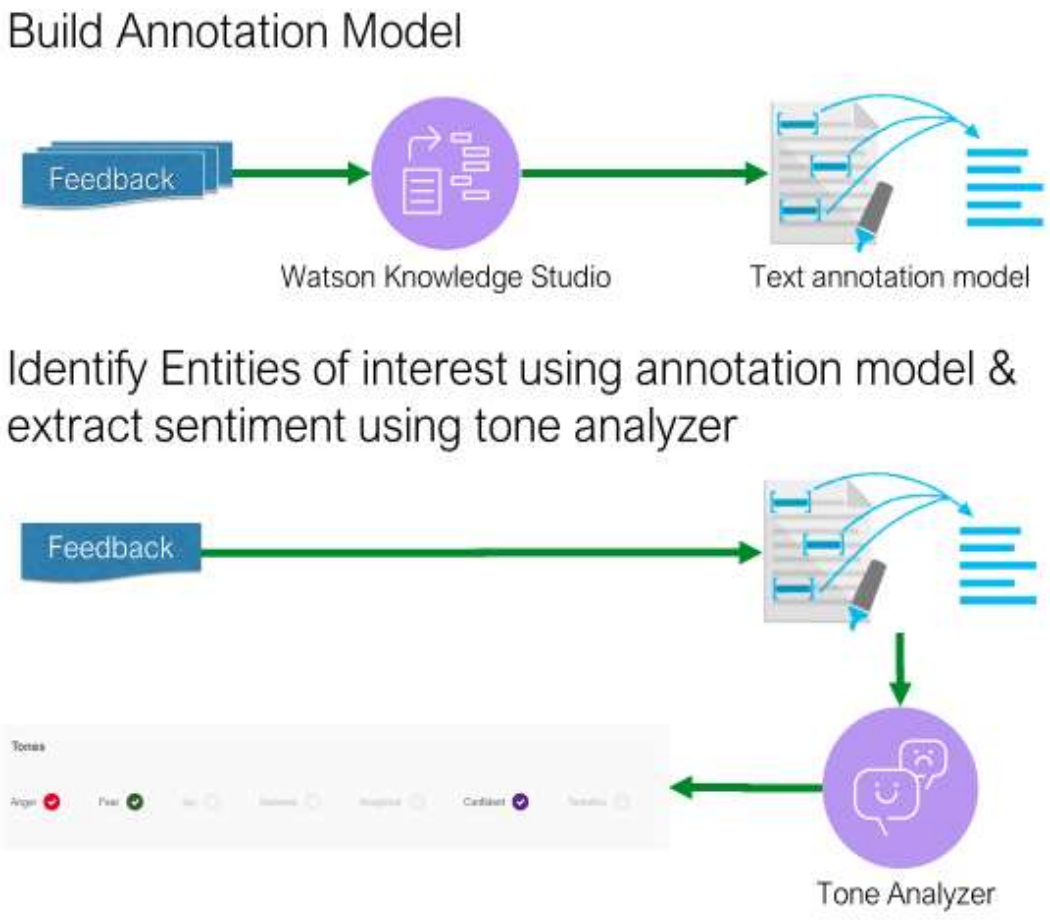
- **Natural language** understanding for building AI models to annotate plain text
- **Tone Analyzer** to understand tone of a sentence
- **Natural Language Classifier** to classify text into pre-determined groups
- **Watson Knowledge Studio** to build custom annotation models
- **Watson Discovery** to search for useful insights from plain text document corpus
- **Watson Studio Local** to build custom models using Pyhton/R/Scala etc.

Solution Architecture



Workshop Hands-on scope

Building the application to orchestrate the overall workflow and provide the user interfaces is outside the scope of this workshop. This workshop will focus on the underlying AI capabilities that the application will use to perform its job. Specifically, we shall focus on two areas as shown below:



Provisioning services in IBM Cloud

Follow the link below to provision services in IBM Cloud. Look at "**Before you begin**" section under each service. Just provision the services at this point. No need to complete the entire tutorial in the link:

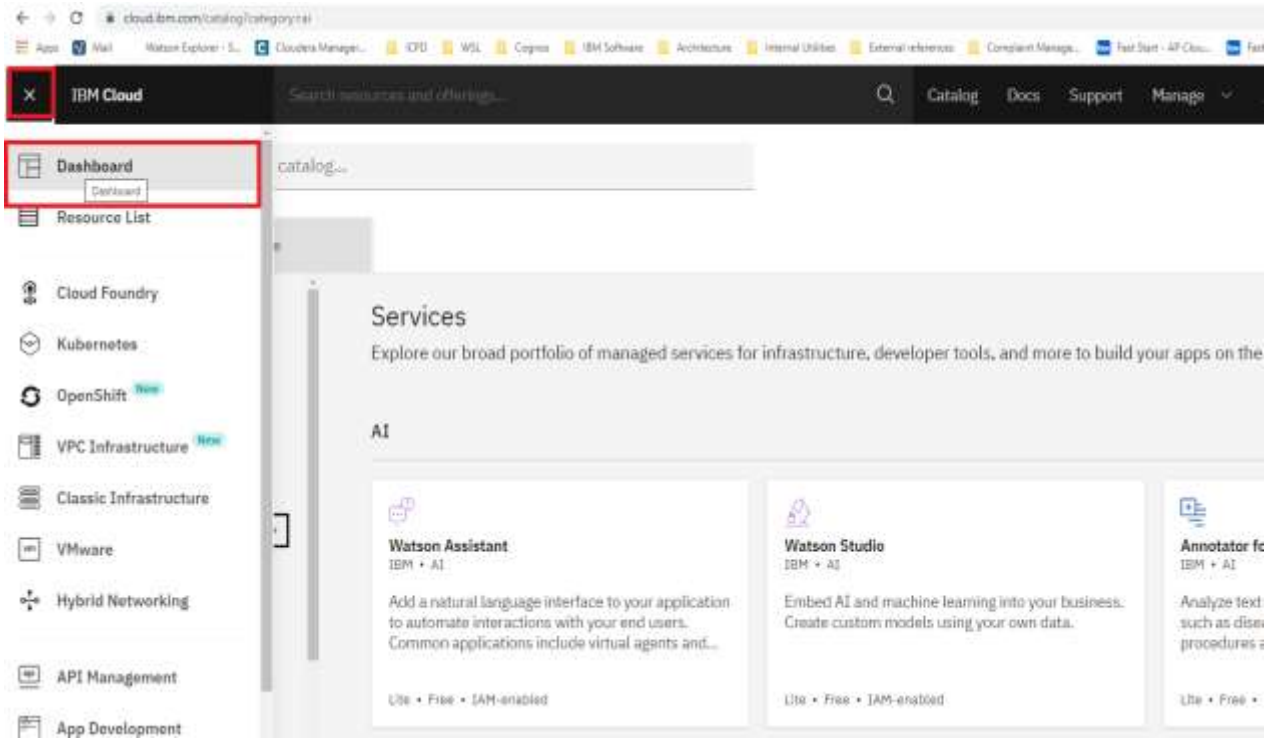
Knowledge Studio:
https://cloud.ibm.com/docs/services/watson-knowledge-studio?topic=watson-knowledge-studio-wks_tutintro#instance
Natural Language Understanding:
https://cloud.ibm.com/docs/services/natural-language-understanding?topic=natural-language-understanding-getting-started#before-you-begin
Tone Analyzer: https://cloud.ibm.com/docs/services/tone-analyzer?topic=tone-analyzer-gettingStarted#prerequisites
Natural Language Classifier: https://cloud.ibm.com/docs/services/natural-language-classifier?topic=natural-language-classifier-natural-language-classifier#prerequisites

If you need a Python runtime then you can also subscribe to Watson Studio and use Notebook IDE with a Python runtime. Once you have provisioned the services, you can go to your Dashboard and see the services you have subscribed to.

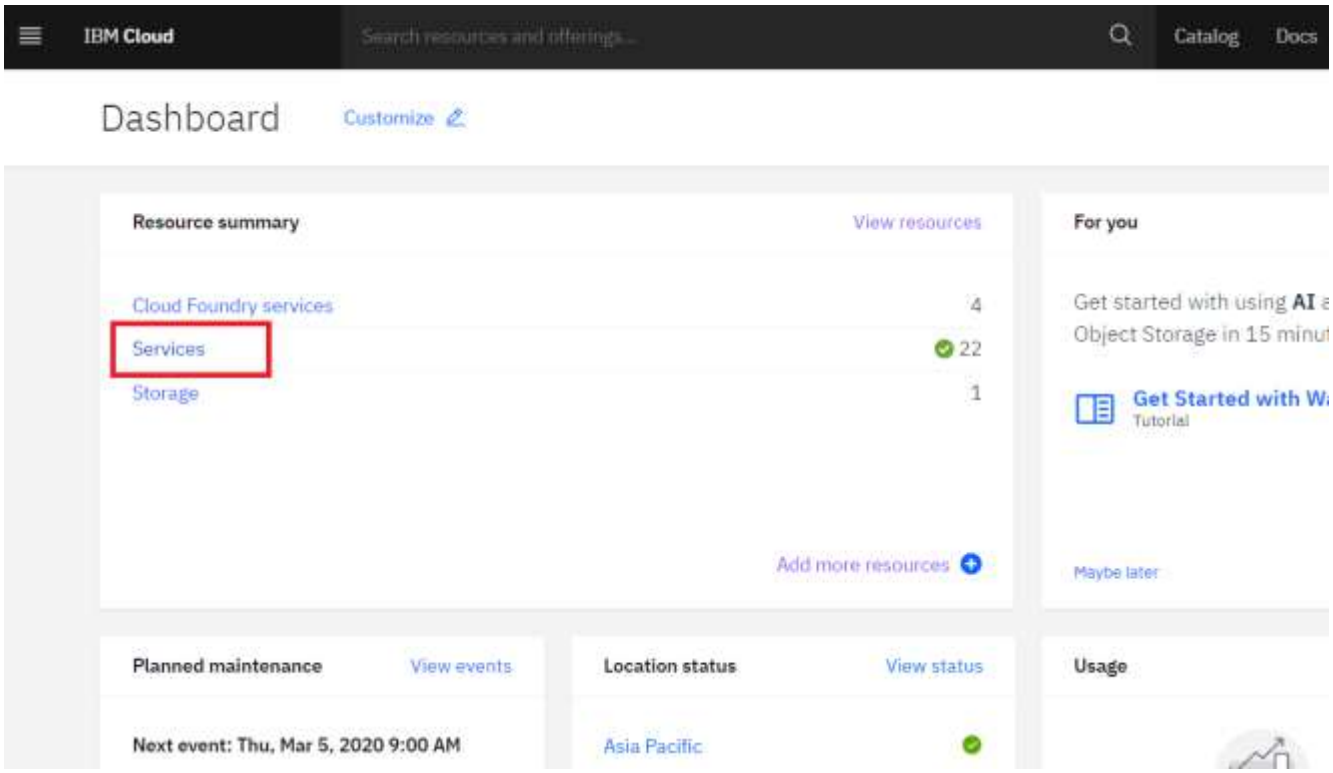
Build annotator model using WKC

1. Launch Watson Knowledge Studio

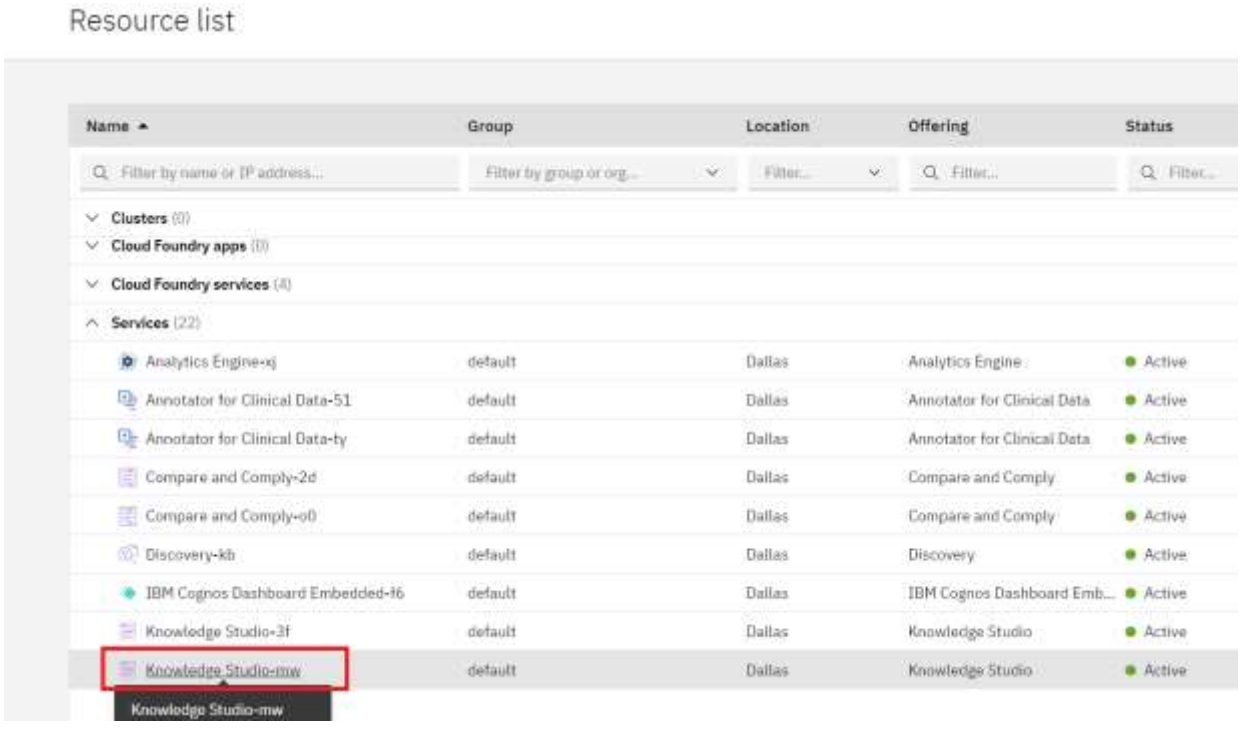
Go to cloud.ibm.com and click on **Dashboard** from the top left ribbon



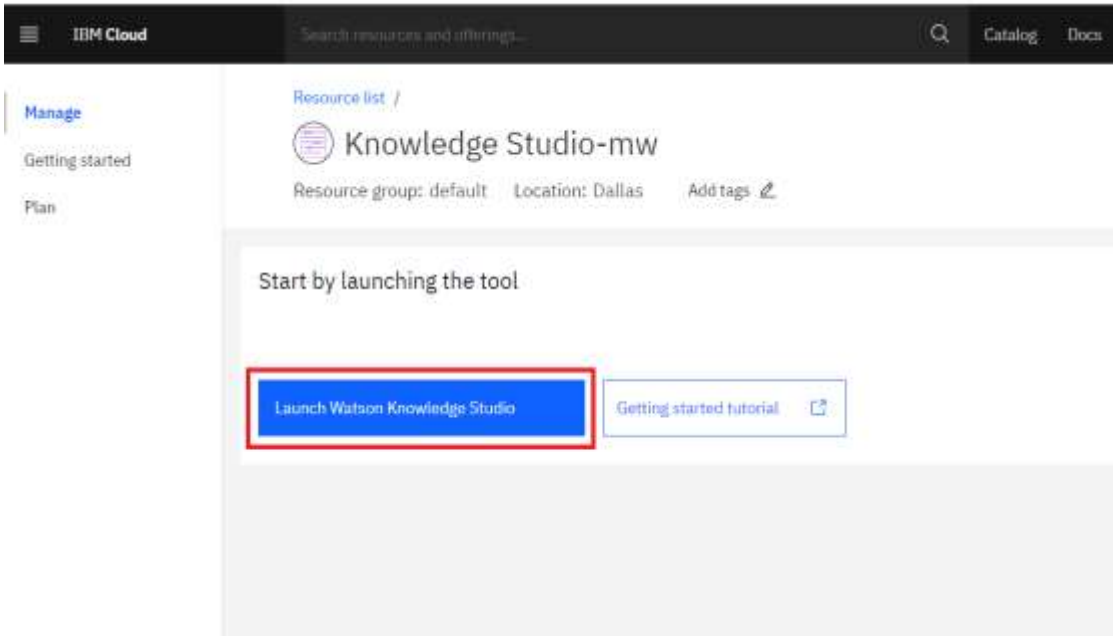
The Dashboard shows the list of **Services**.



Click on **Services** to see your subscriptions and then click on your **Knowledge-Studio**-xx instance

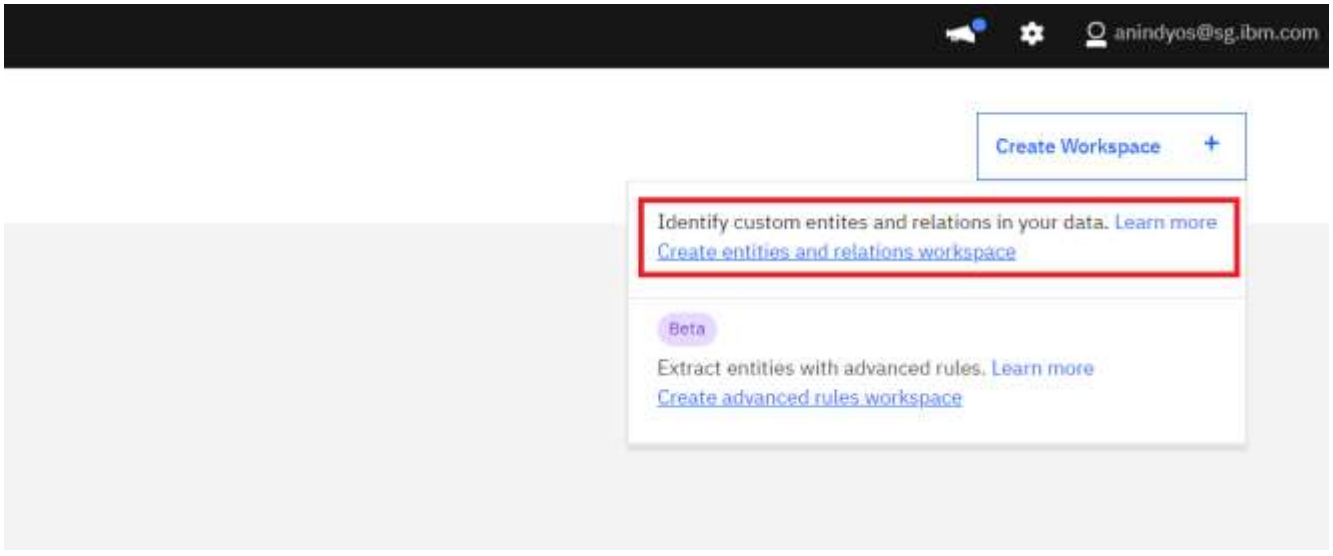


Knowledge Studio launch page opens. Click on **Launch Watson Knowledge Studio** to open Knowledge-Studio.

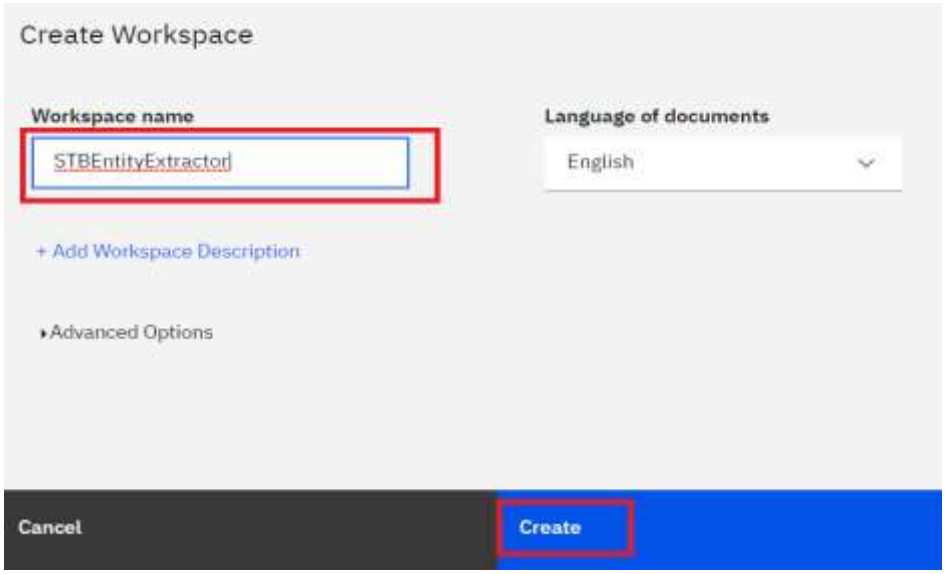


2. Create Workspace

Create a workspace by clicking on **Create Workspace** on top right and choosing the top option of Create entities and relations workspace.

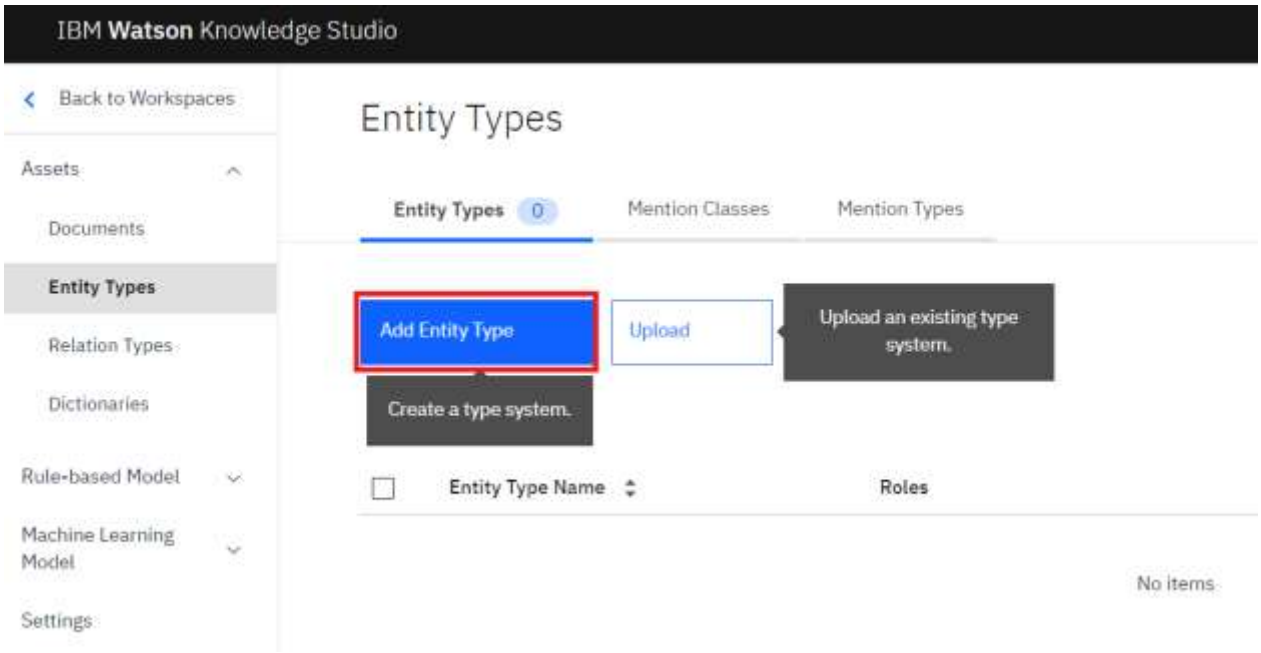


Give it a proper name like STBEntityExtractor.

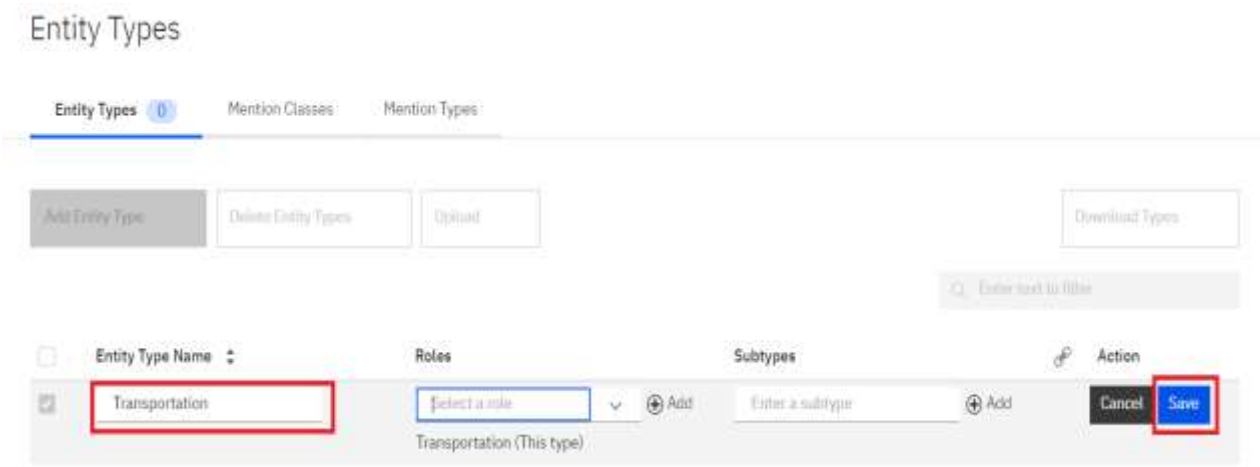


3. Create Entity Types

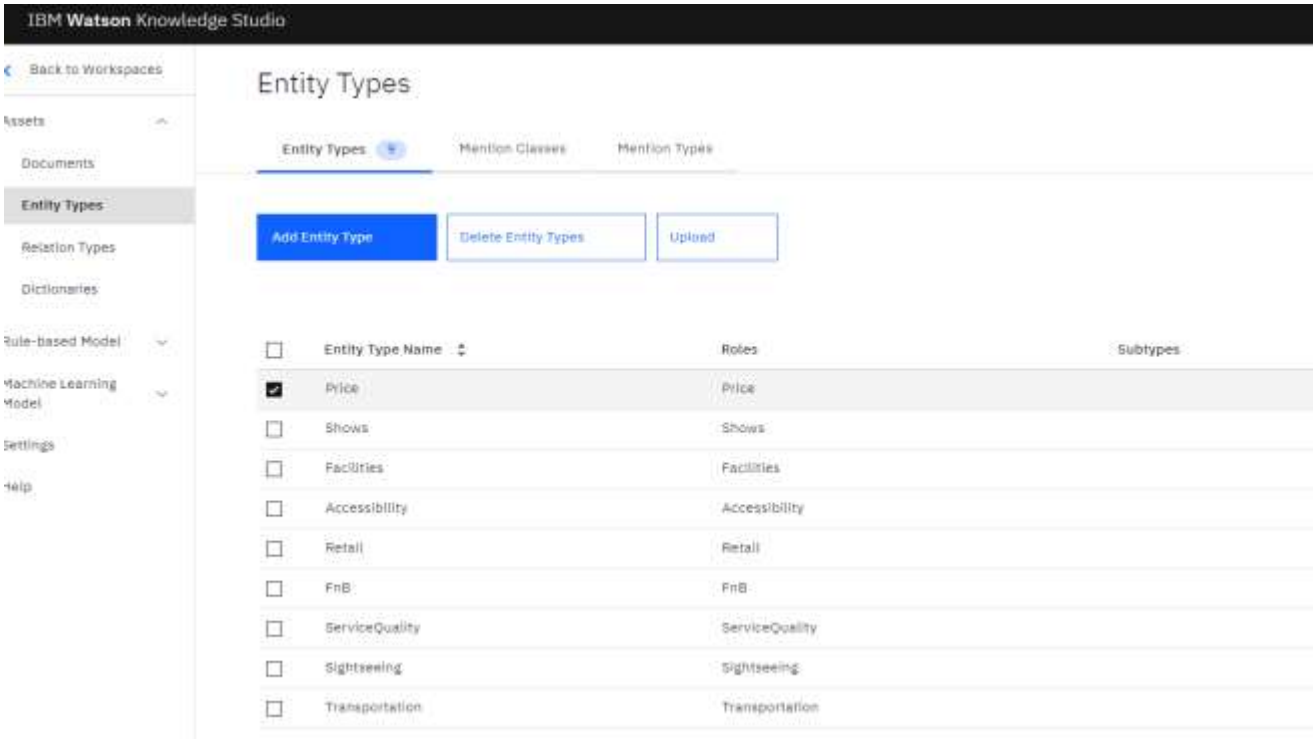
To create Entities, click on **Add Entity Type**



Enter the entity type value and click on **Save**



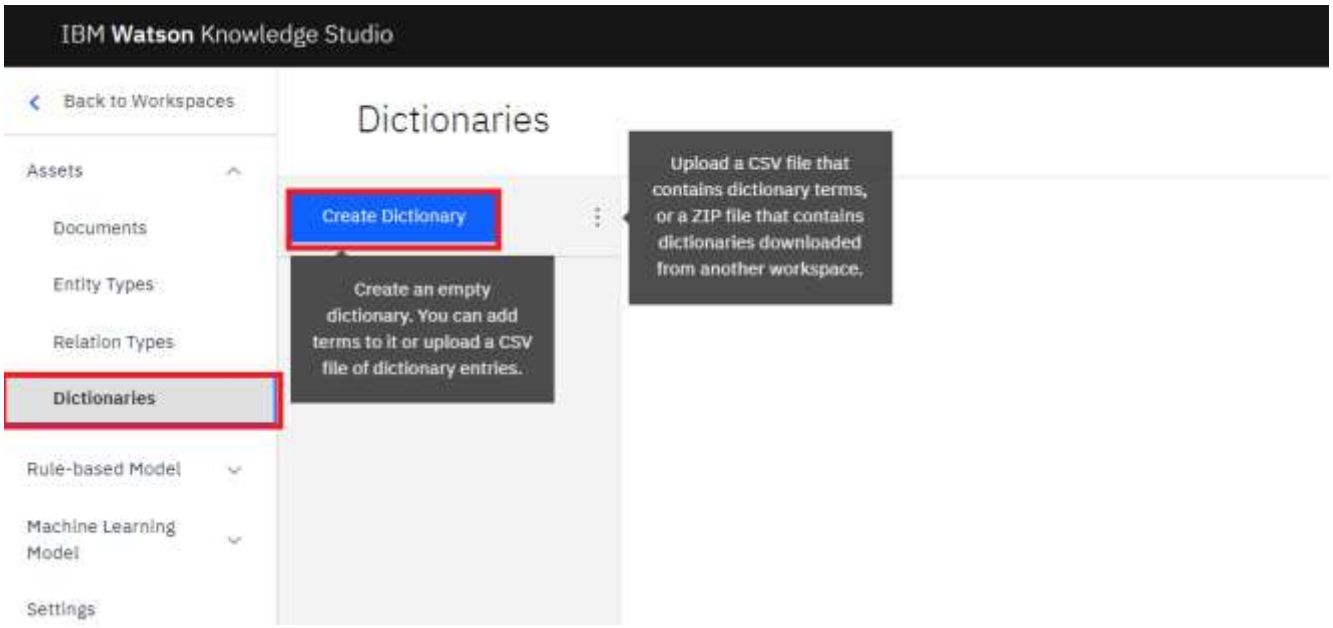
Repeat this operation for all entity types



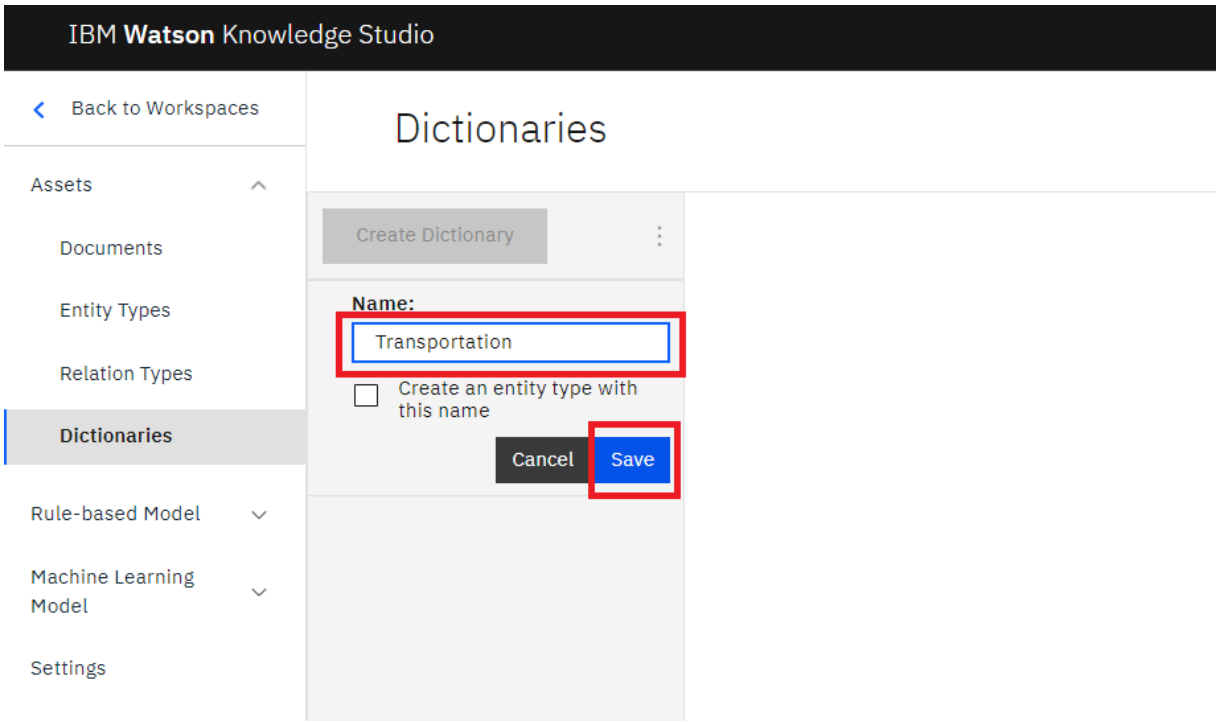
4. Create Dictionaries

Though not mandatory, to help your model to accurately identify your entities you may define dictionaries and associate them with your entities. This helps when we are focusing on a narrow or finite set of values and using those values to identify the entity, which is mostly the case in this scenario. Hence we shall define dictionary for most of our entities.

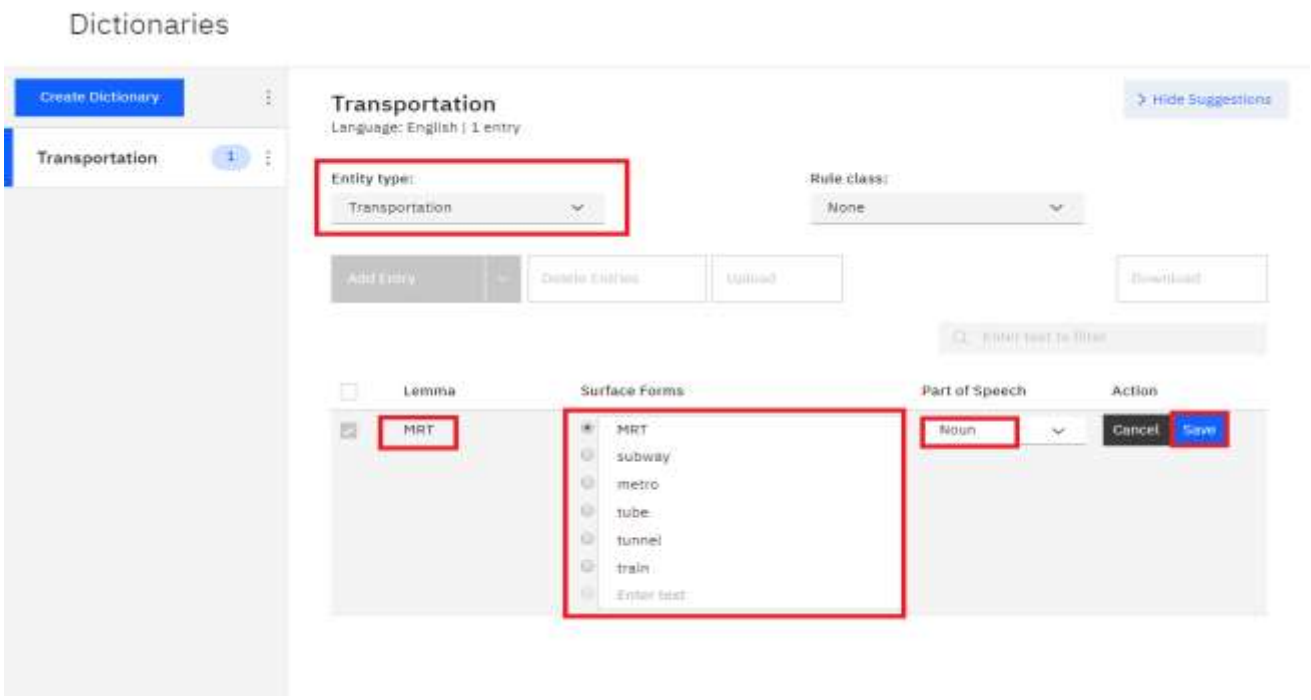
Click on **Dictionary** and **Create Dictionary**



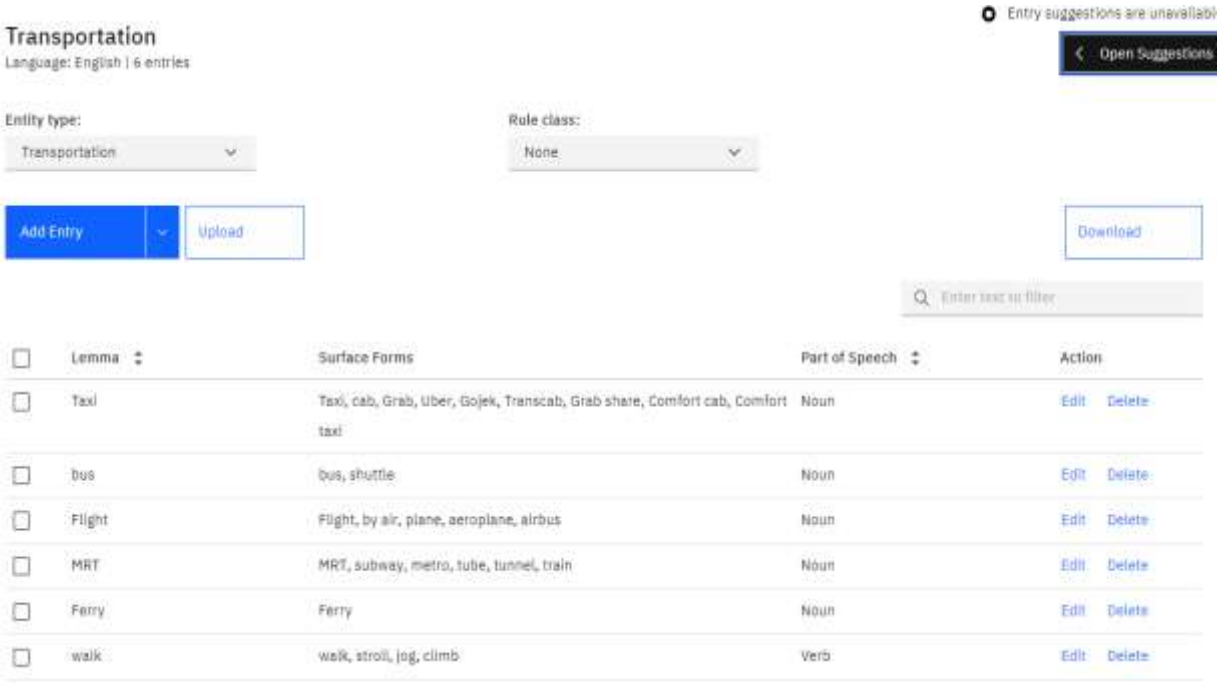
Click on **Create Dictionary** to create a new dictionary. Name it “Transportation” and **Save** it.



Associate Entity Type “Transportation” to this Dictionary. Create Lemma forms example MRT and fill up Surface Forms as shown below. Click Save to save the entry.



Repeat the above step to create other Dictionary items under “Transportation” Dictionary as shown below:



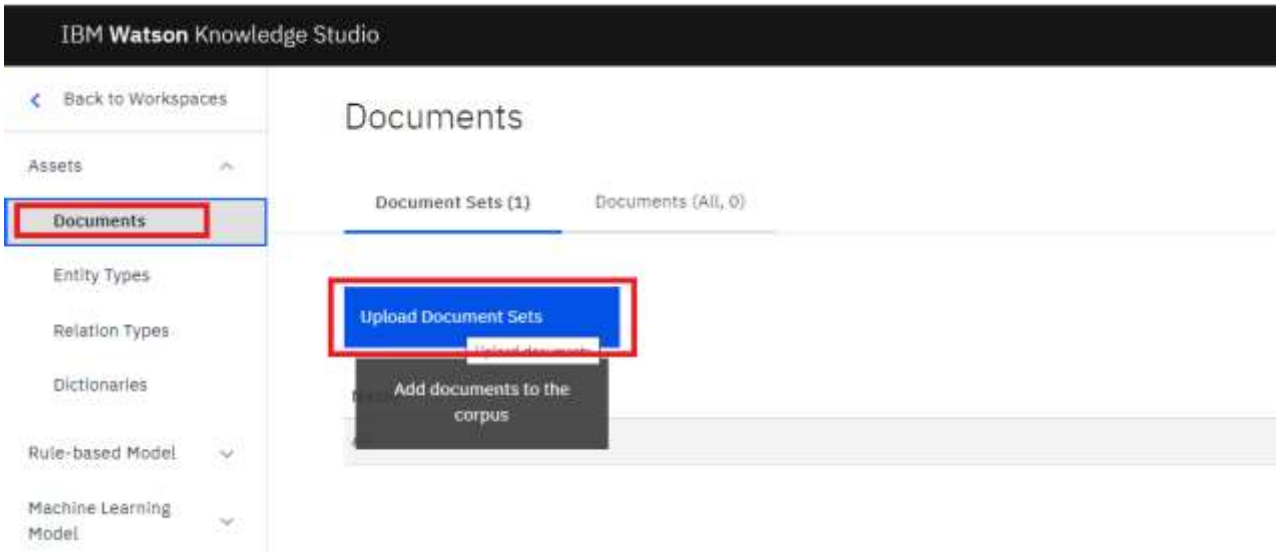
Similarly create Dictionary for the remaining entities

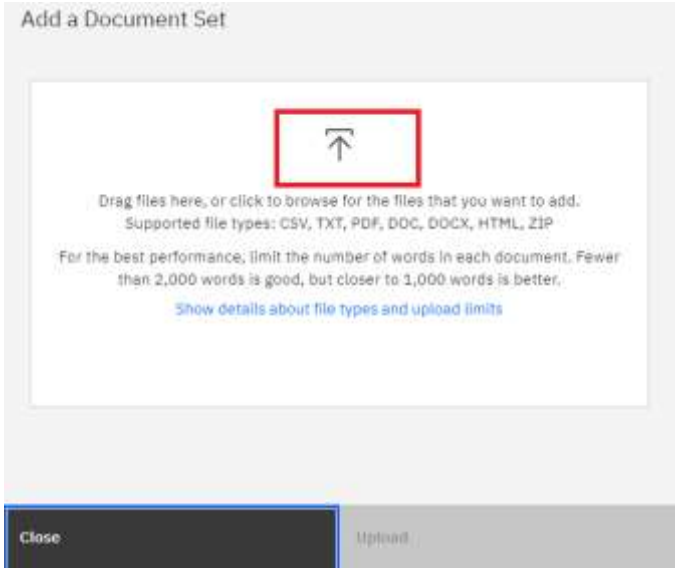
Sample dictionaries created for some of the above entities are attached herewith for reference.

FnB_1583211748859.csv Shows_1583211328452.csv Facilities_1583211245299.csv Retail_1583210878563.csv Price_1583212757888.csv

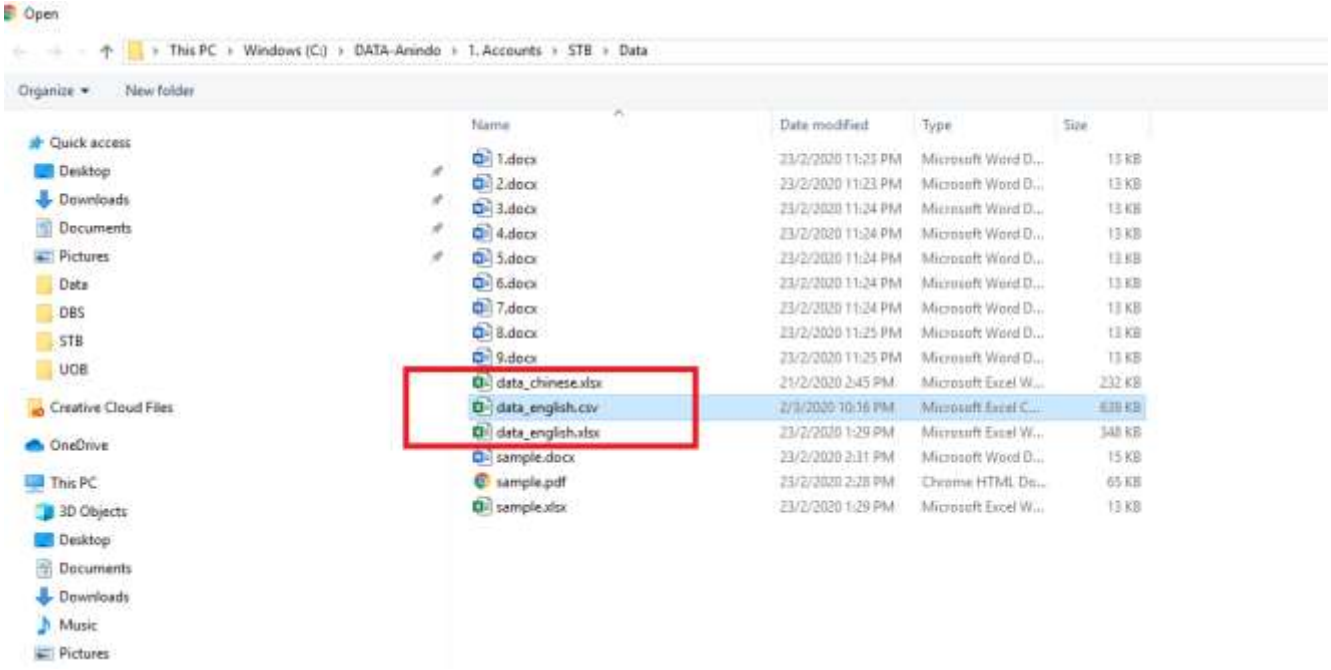
5. Upload Training data set

After all dictionaries have been created, upload the training data set as a csv file. Click on **Documents** and **Upload Document Sets** to upload

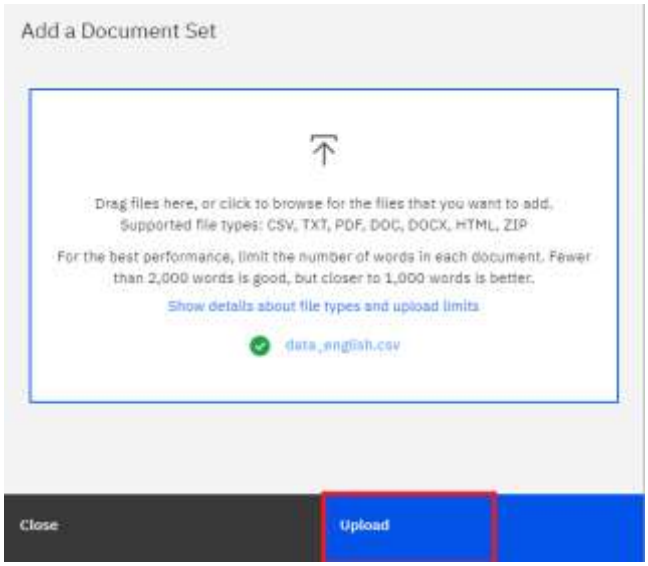




Select the csv and open it.



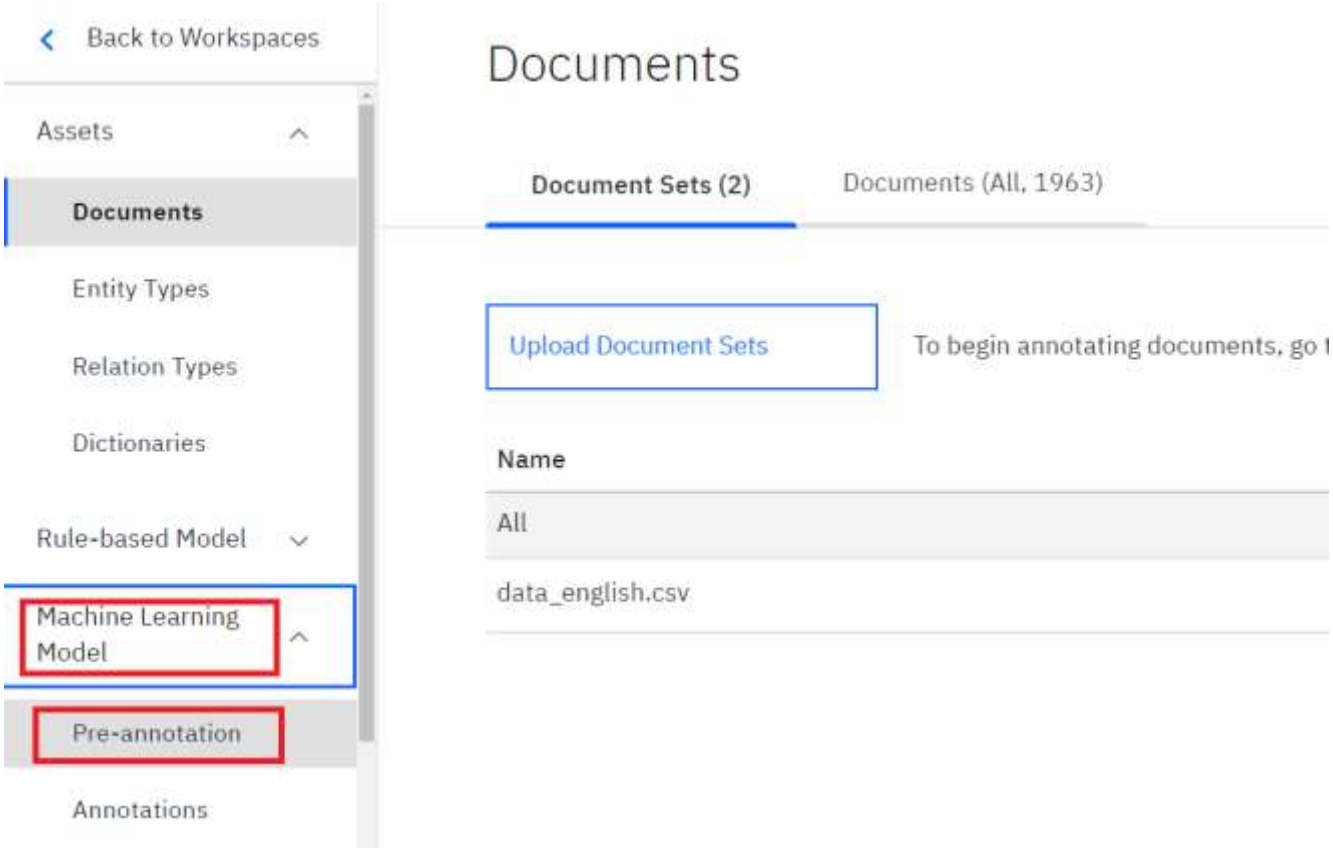
Click on **Upload.** Wait till upload completes. There are resource limitations in the free account. Hence use a small sample as this workshop is for functionality illustration only. You may not be able to upload large number of files/documents/rows. Select first 30 rows of the document and create a subset csv file that we shall use for this exercise.



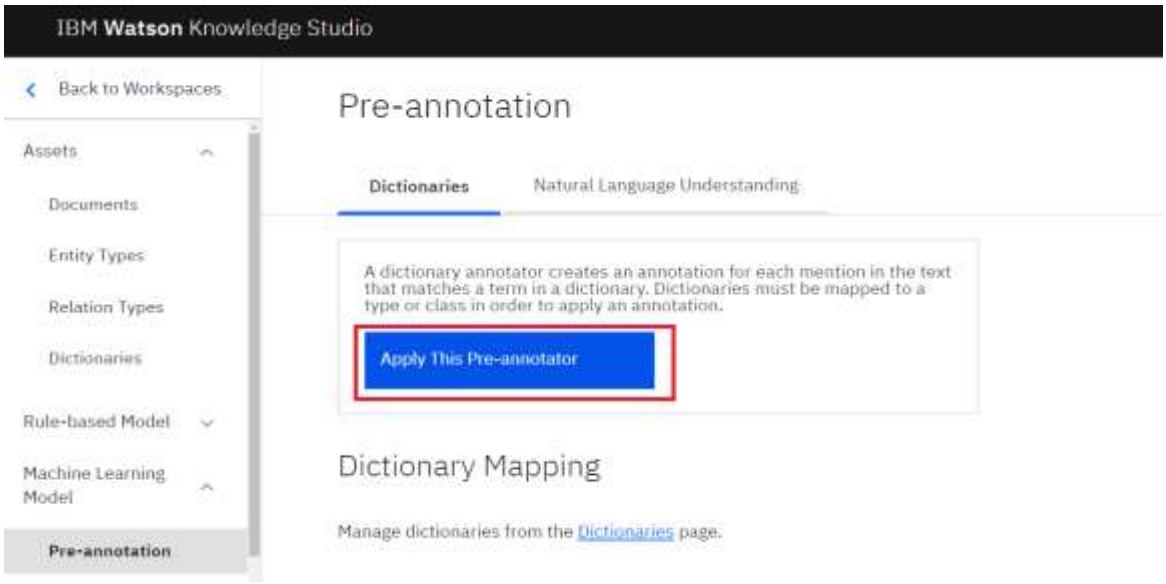
Verify that document set is uploaded properly.

6. Annotate

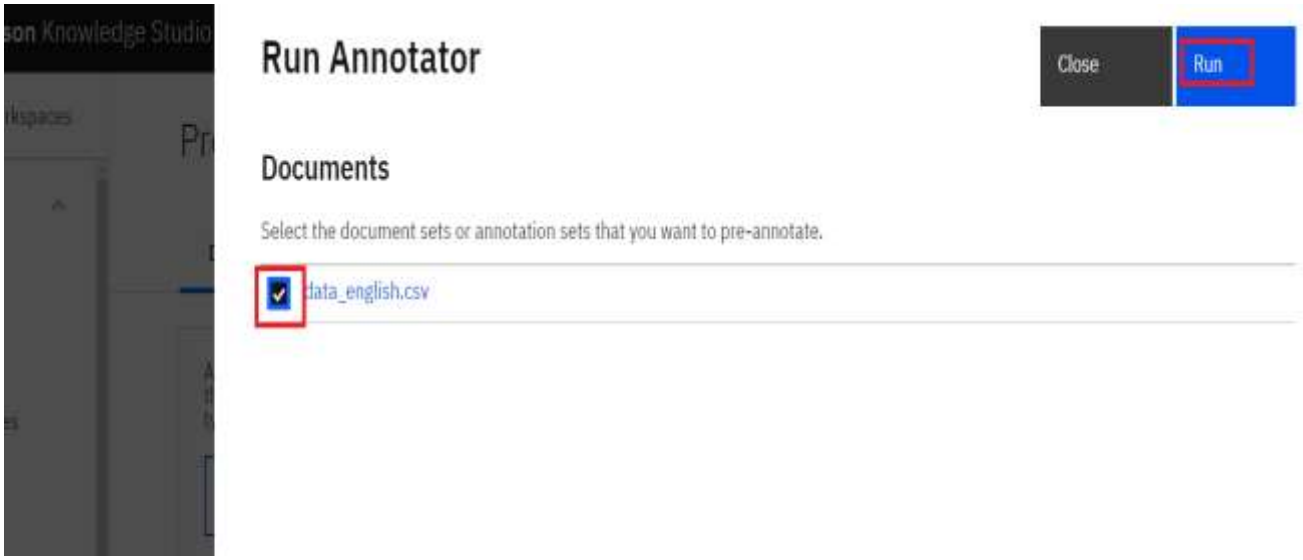
First we would pre-annotate the documents based on the dictionaries that we have defined. To do that go to **Machine Learning Model** -> **Pre-annotate**



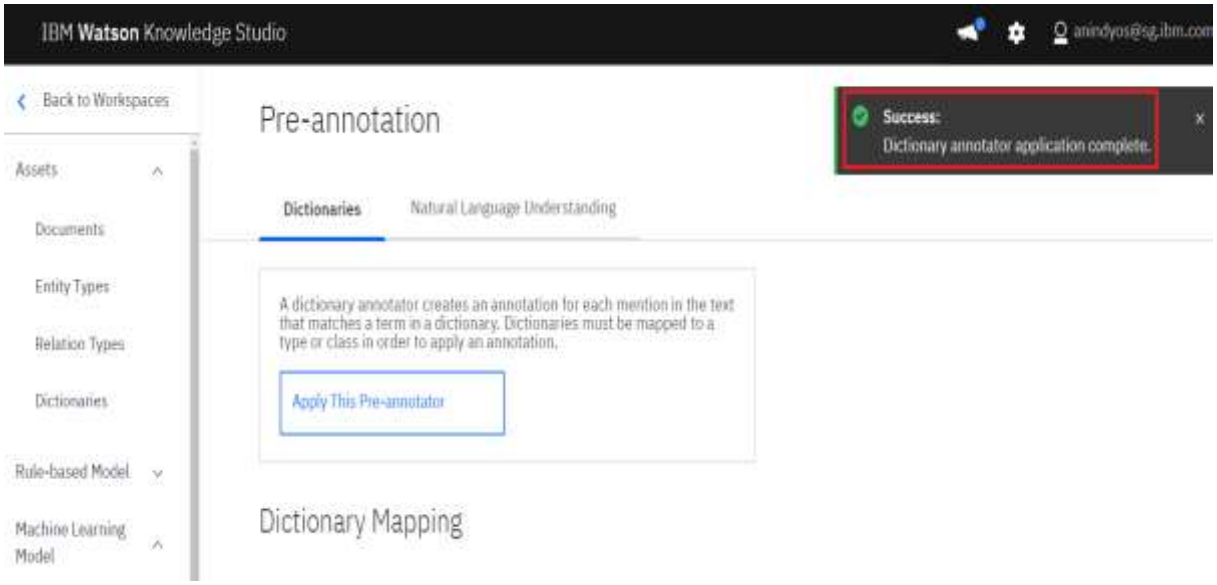
And apply the pre annotator



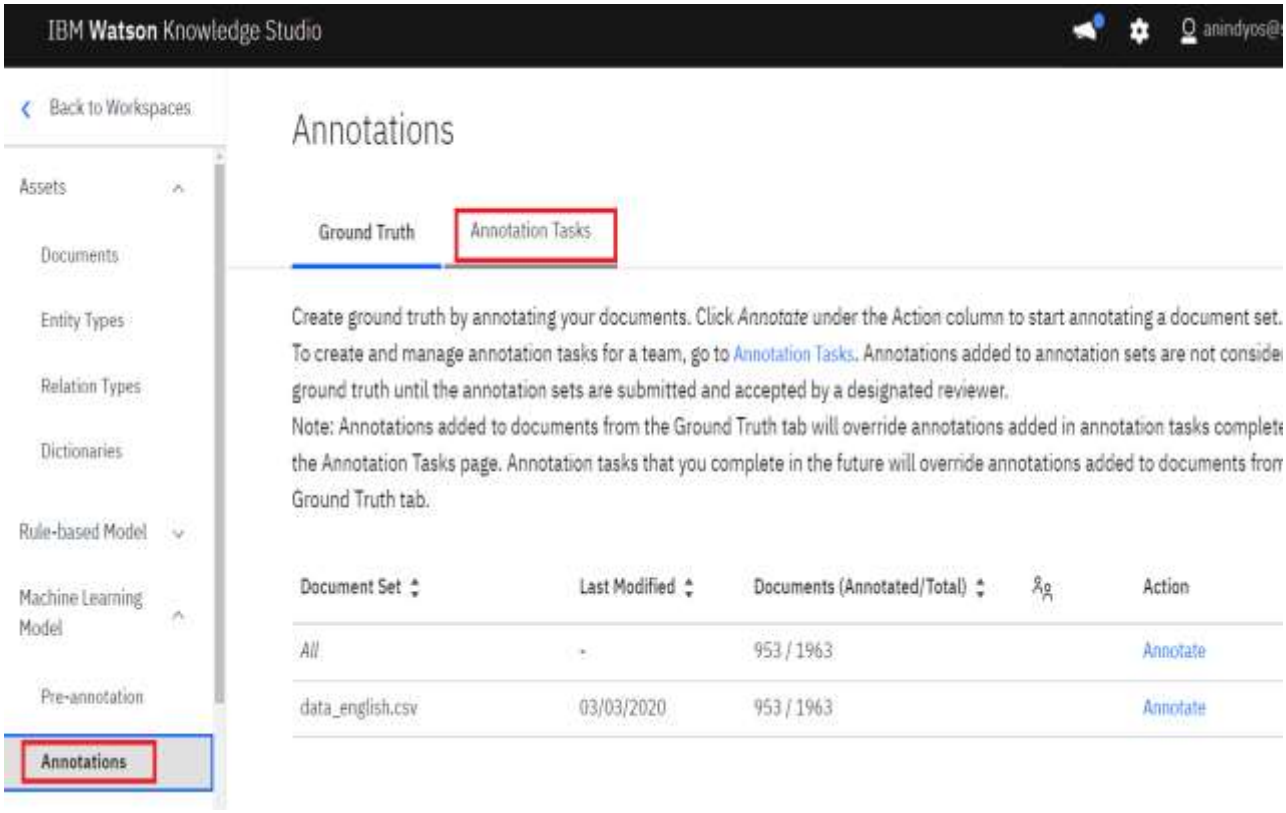
Choose the document set and click on **Run**



Wait till it completes and you get a **Success** message.



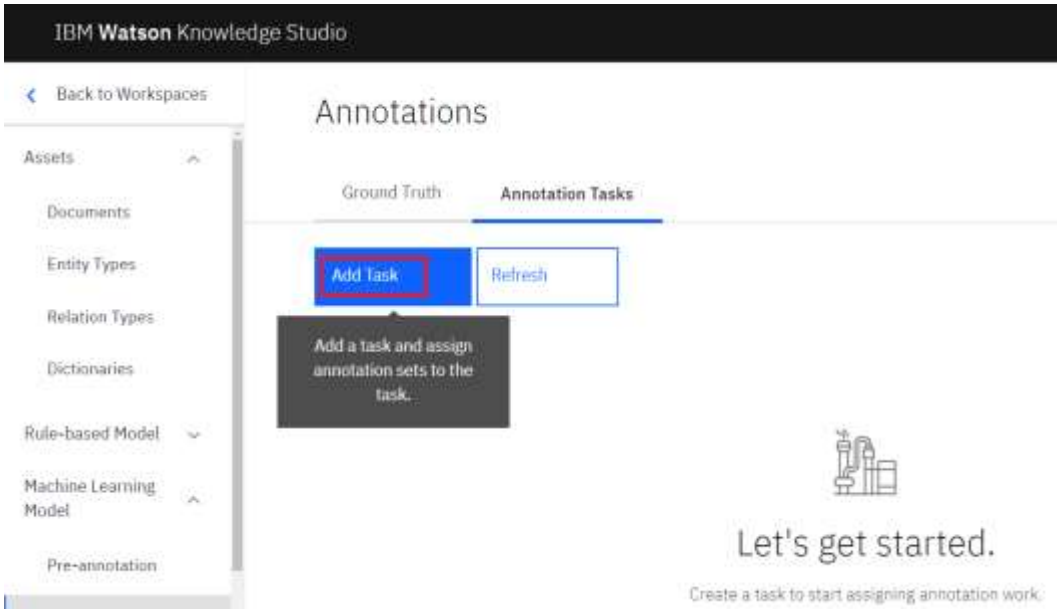
Once this automated pre-annotation is done we shall perform some manual annotation. Click on **Annotations**. You can see the Ground truth. The ground truth shows the training set with dictionary-based annotator applied, without any manual annotation. To perform manual annotation, we need to setup an **Annotation Task**.



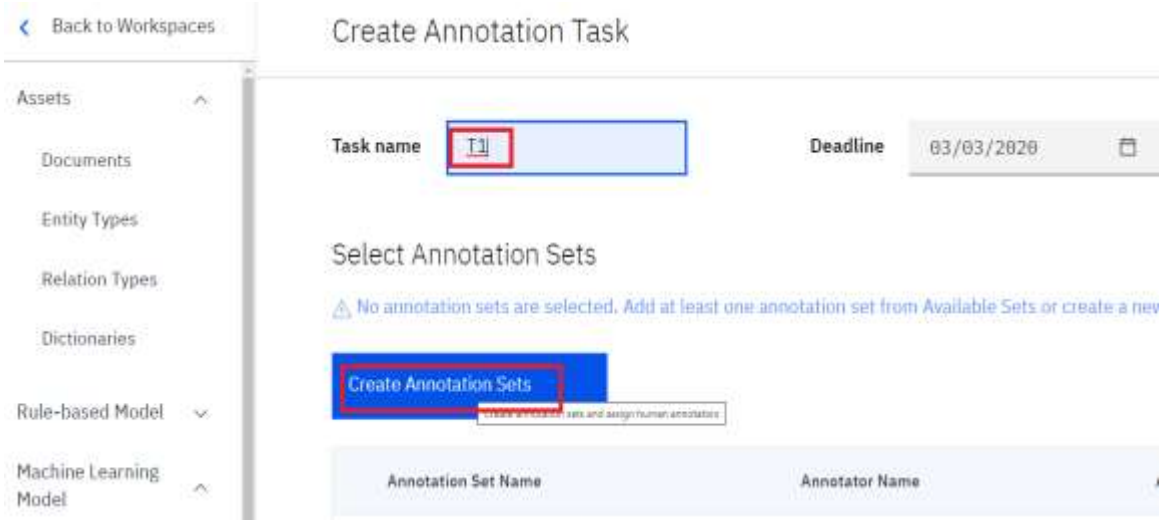
Annotation tasks can be performed in a group, distributed to different team members with some overlapping content. The purpose of the overlap is to ensure that human error is identified, reviewed and corrected. Any document that is annotated differently by different annotators can be flagged by system as a discrepancy and can be sent for review. Once review is completed, it can be **Accepted** or **Rejected** and sent back to the team. All the annotation tasks need to be **Accepted** before it can be submitted for training.

For our scenario, we shall use a single user annotator instead of team annotation. We shall create one Task and assign it to one user. The user shall complete all the annotation by himself and submit for review.

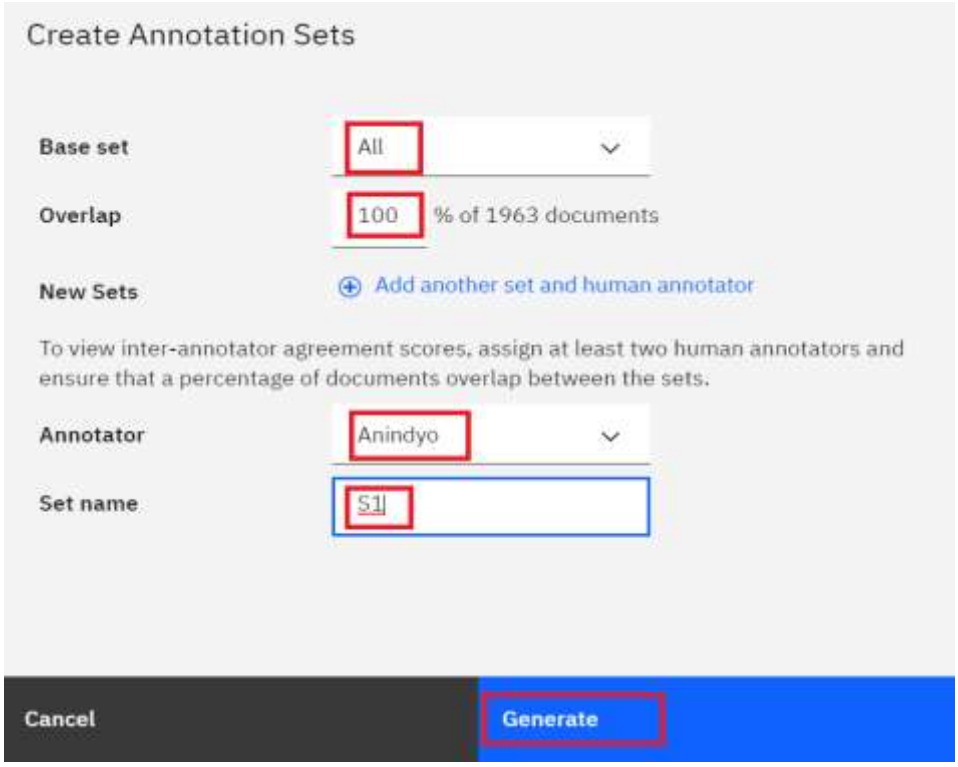
To create Annotation Task click on **Annotation Task** under **Annotations** and then on **Add Task**.



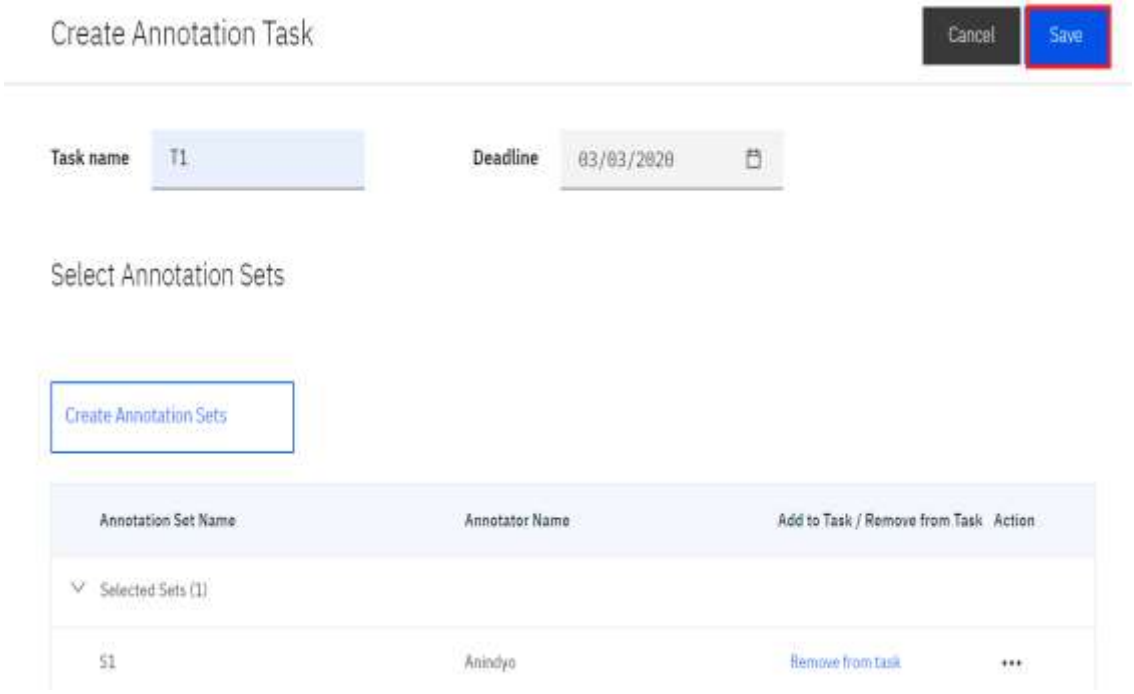
Give the Task a name and **Create Annotation Set**



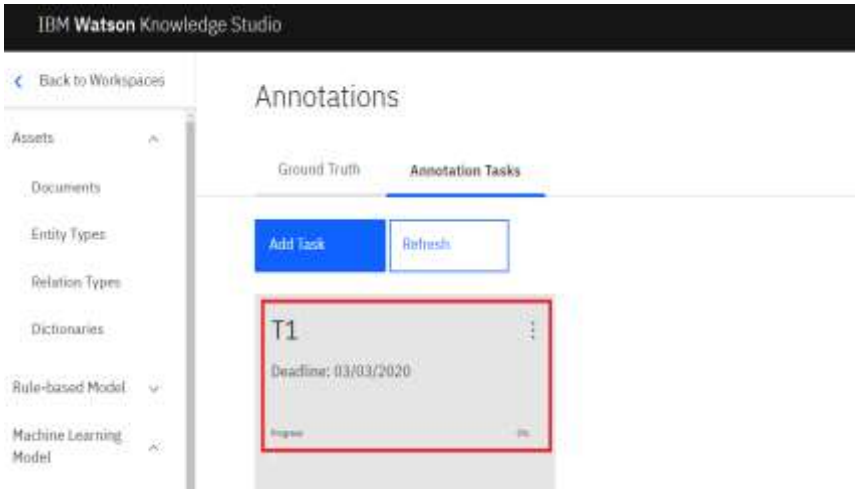
Fill in the details, specify an annotator (you can use your name) and give the Set a name and click on **Generate**.



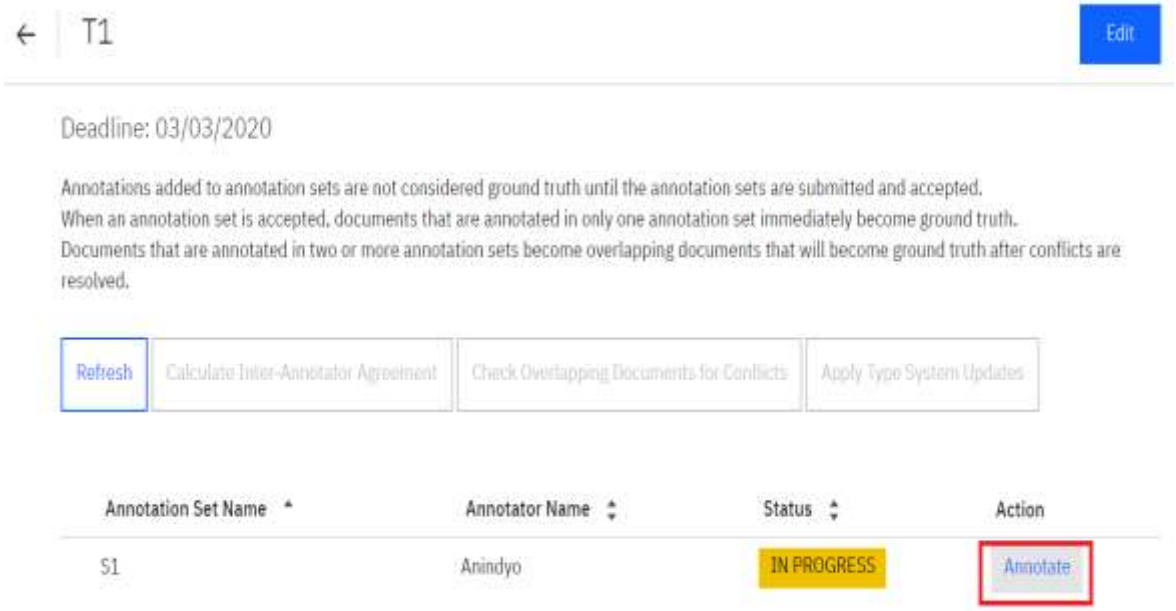
The annotation set gets created. You can **Save** the annotation task now.



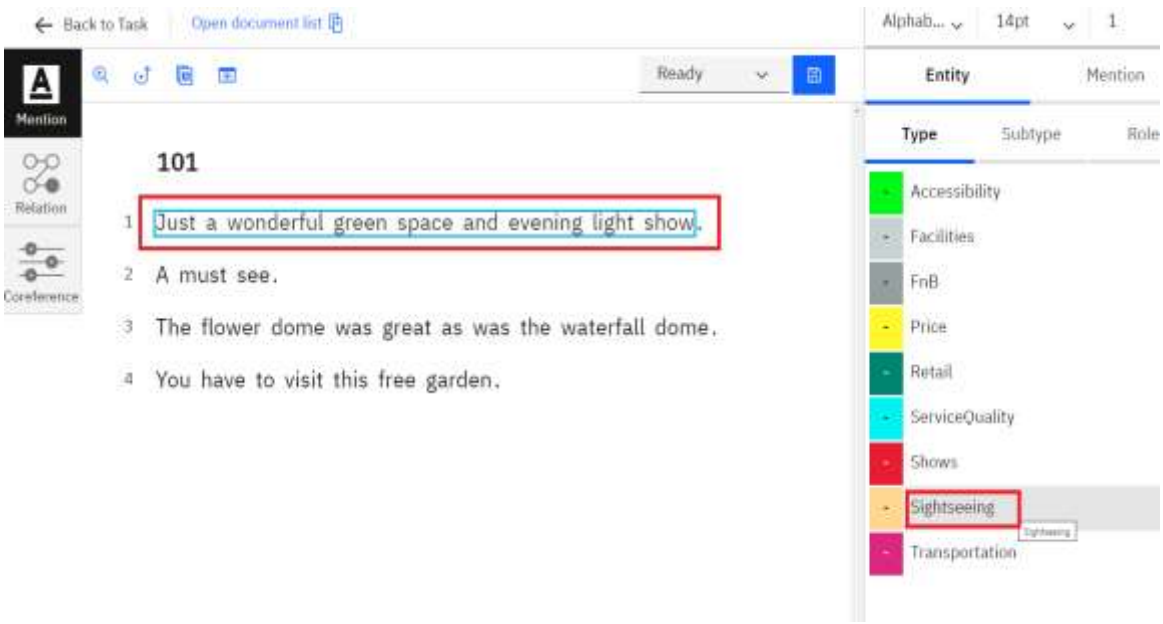
Click on the **Task** to start annotation



and click on **Annotate**

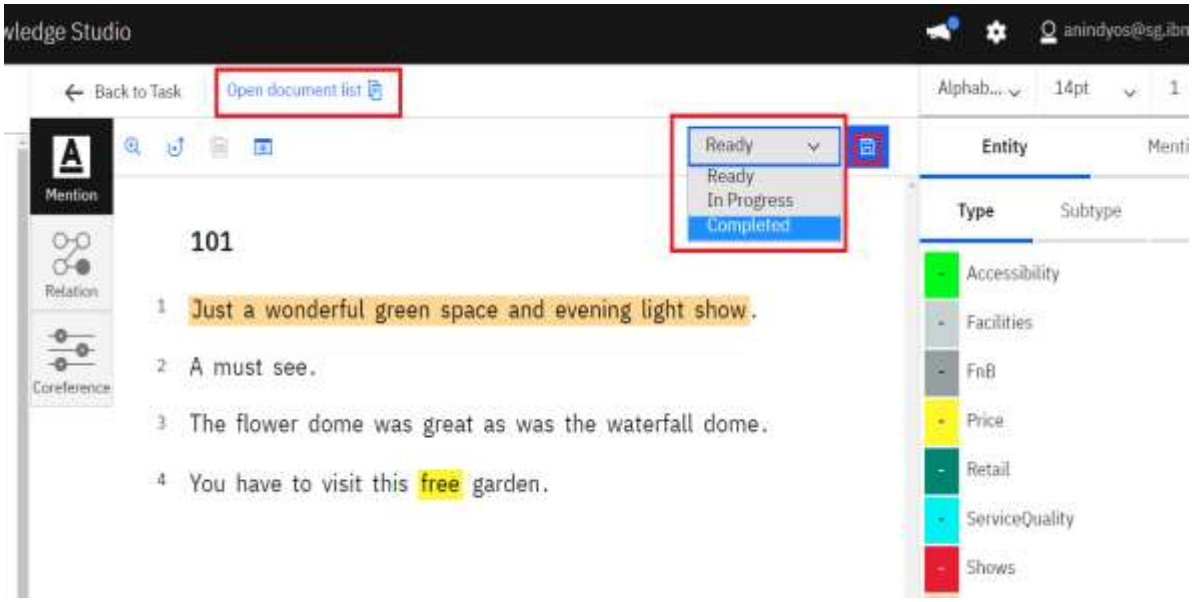


Open a document to annotate. On the middle section of the canvas you can see the document and on the right panel are the entities. We can mark the section of the document that we think matches an entity representation



And click on the corresponding entity to annotate that text with the corresponding entity. The section of the text will be highlighted with the color code of the entity

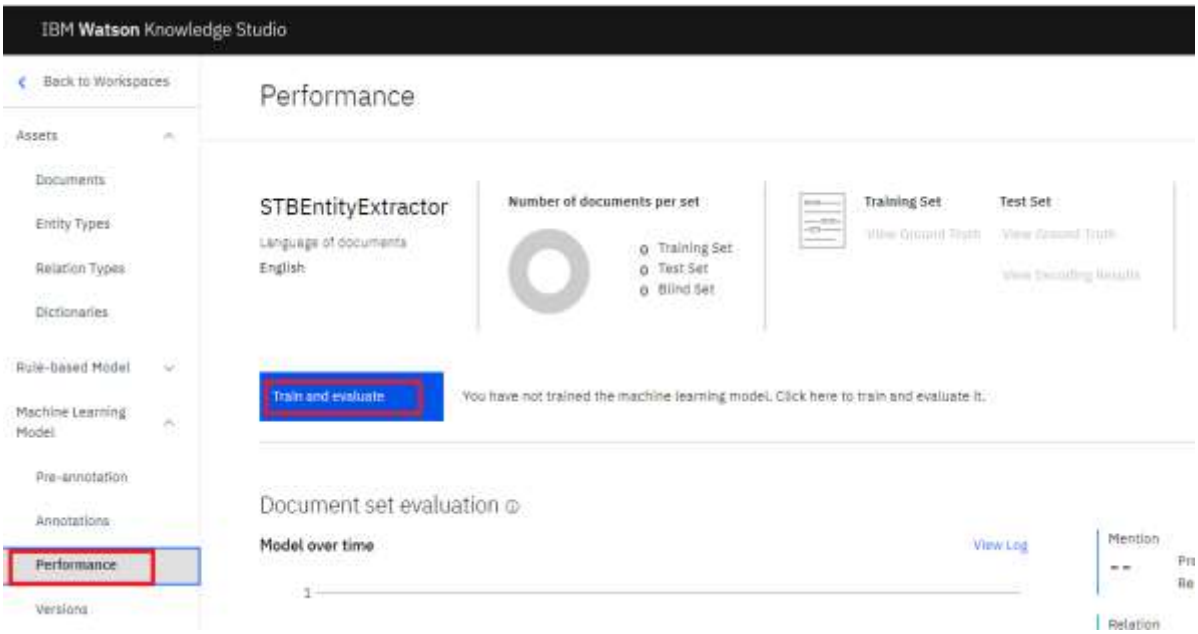
After you have completed annotating the document., change status to **Completed** and save the document



Perform this exercise for all the documents. You would notice that dictionary-based annotations are already done, and no action is required for those.

7. Train & Deploy model

After accepting the task, go to **Performance** under **Machine Learning Model** to train the model.



Follow instructions as shown in the link below:

https://cloud.ibm.com/docs/services/watson-knowledge-studio?topic=watson-knowledge-studio-wks_tutml_intro#wks_tutless_ml8

8. Deploy Model

9. Test Model using API

To extract entities, your application shall call the NLU (Natural language Understanding) API instance where we have deployed the annotator model. In this workshop since we don't have the application, we shall test the same using curl commands. You can use Python or java etc. to test the same as well if you have the corresponding runtimes.

*** To Get Entities, run the below curl command from your laptop command prompt

curl --user "apikey":"*<Your API Key>*" "<*Your NLU instance url*>/v1/analyze?version=2019-07-12" --request POST --header "Content-Type: application/json" --data @*<Your json file name with fully qualified path example C:\Users\AnindyoSarkar\Documents\transportfeedback.json>*

Sample json file attached below for reference:

The output may not be always 100% accurate as our training set is limited. We would need to train the model with a larger training set which would be representative of the production data to get more accurate results. Normal data science life cycle should be established and deployed

which would review the accuracy at regular intervals and train model continuously to keep up the accuracy of the deployed model

Get Sentiments of entities

Once the transactions with entities of interest are identified, the custom application is expected to store this information in a json database like Mongo DB for further processing.

The next step of the process would be to identify sentiments associated with specific entities. For the same we shall use Tone Analyzer API in this workshop. The sentence or entire document which has the identified transaction shall be submitted to the Tone Analyzer API and sentiments retrieved. The custom application would then join this sentiment with the identified entity and save the result in a relational DB.

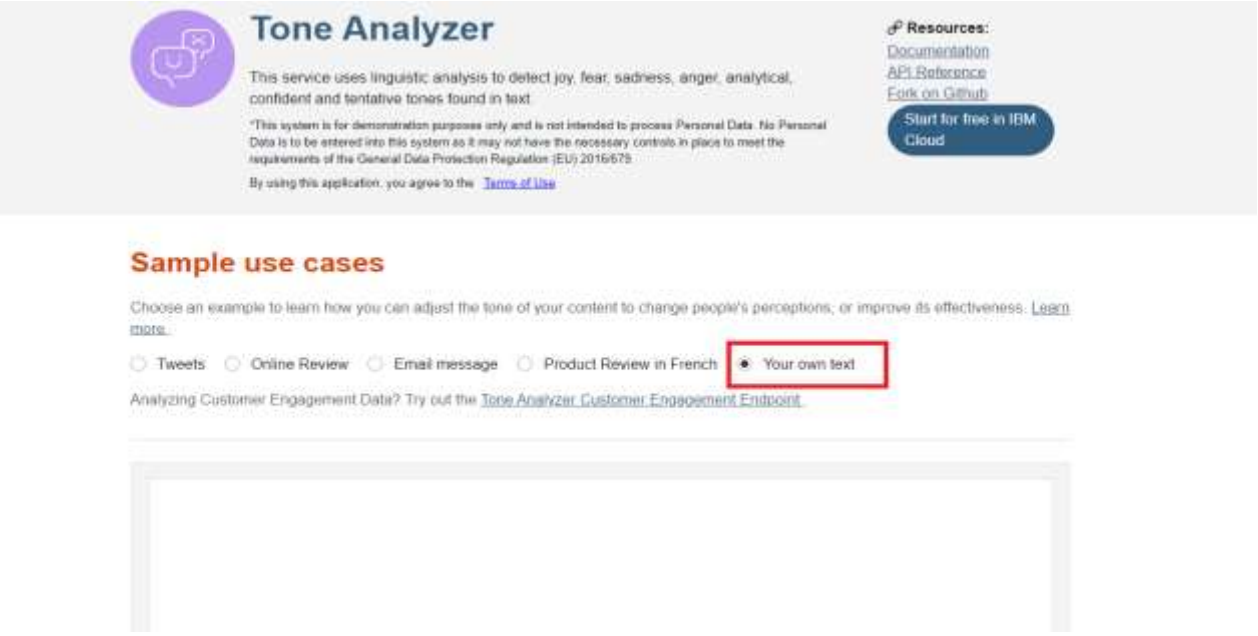
1. Use demo application

For our workshop lets use the demo application to test the tone of the feedback. Click the link below to access the application:

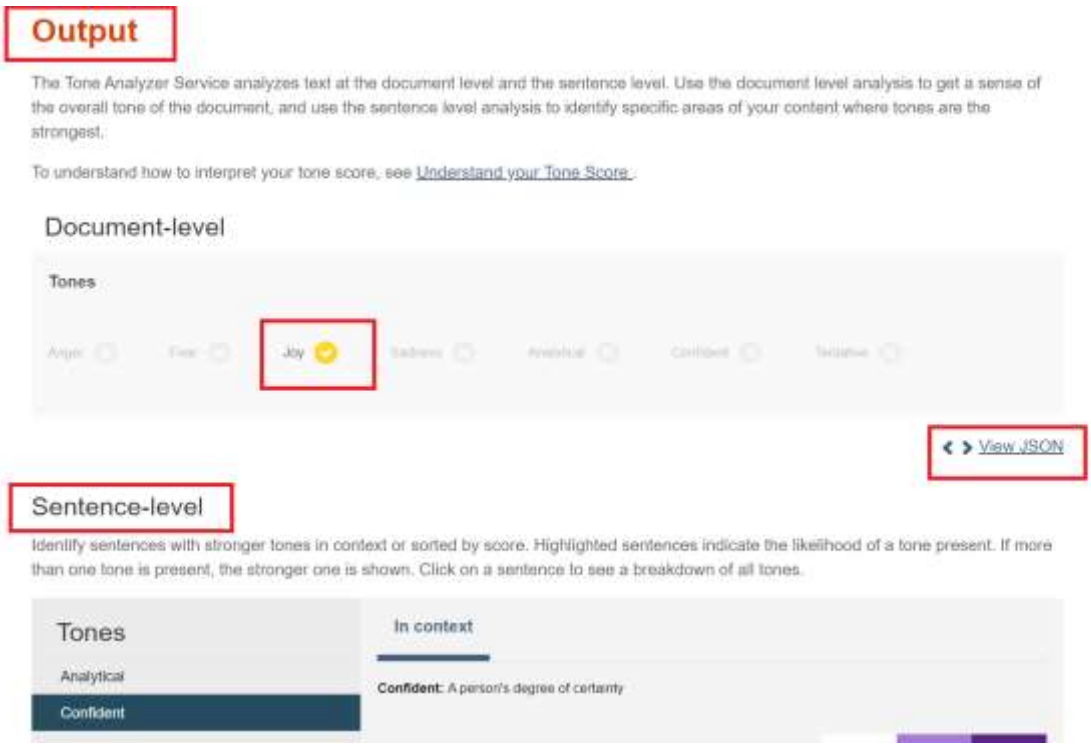
https://tone-analyzer-demo.ng.bluemix.net/?_ga=2.167287678.1795414099.1583208908-1313120485.1583208908&cm_mc_uid=62290322875615819138240&cm_mc_sid_50200000=99633231583227288022&cm_mc_sid_52640000=99799621583227288067

or google "**Watson Tone Analyzer demo**" and access the application from the result returned.

Choose "**Your Own Text**" in the same use cases



Paste any document content from our sample and click on **Analyze**



Review the overall output. You can view the json and see the individual scores.

You can also review the sentence level feedback and identify sections of the feedback where that sentiment was detected.

2. Call API

You can achieve the same result by using the Tone Analyzer API instead of using the demo application. For detailed instructions use the link below:

https://cloud.ibm.com/apidocs/tone-analyzer

This brings us to the end of our workshop.

The End