**Name: C Bhuvaneshwari**

**College** **code:** 9530

**College name:** st.Mother Theresa Engineering College

**Team id:**

**Naan Mudhalvan id**:au953021104008

**Project Name:** Image Recognition with IBM cloud

**Problem Solution:**

**Consider incorporating sentiment analysis to generate captions that capture the emotions and mood of the images**

**Sentiment analysis is used ubiquitously to gain insights from text data. For example, by using sentiment analysis, companies can understand the voice of the customer or the market sentiment of the company. However, due to the lack of a standard infrastructure and standard libraries, many sentiment analysis projects remain at the proof-of-concept (POC) level and are never put into production.**

**With IBM Watson NLP, IBM introduced a common library for natural language processing, document understanding, translation, and trust. IBM Watson NLP brings everything under one umbrella for consistency and ease of development and deployment. This tutorial walks you through the process of using a pretrained model for sentiment analysis as well as fine-tuning a sentiment analysis model using the watson\_nlp library.**

**The watson\_nlp library is available on IBM Watson Studio as a runtime library so that you can directly use it for model training, evaluation, and prediction. The following figure shows the IBM Watson NLP architecture**

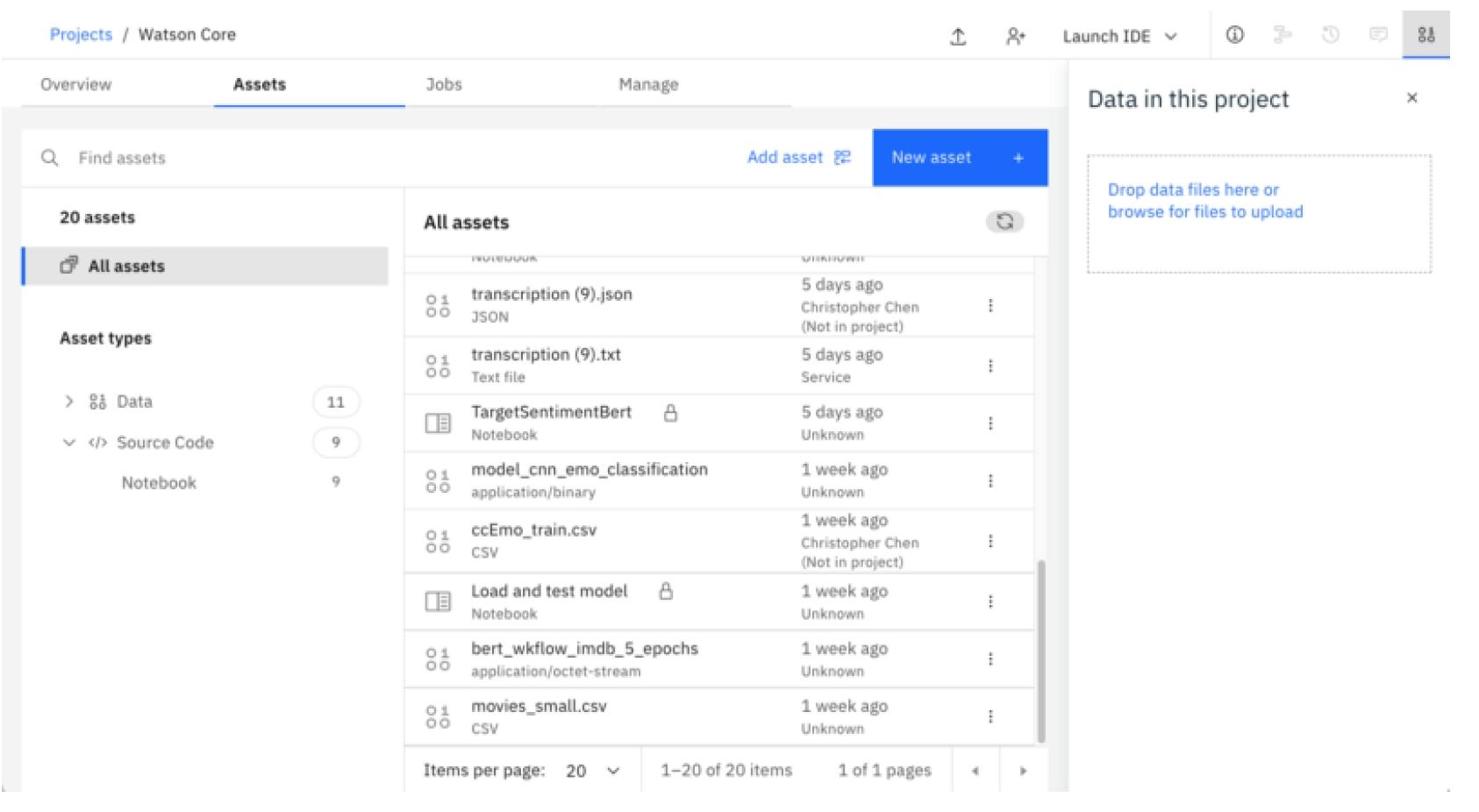
**Steps**

**The steps in this tutorial use an example of IMDB movie reviews from Kaggle to walk you through the process.**

**Step 1. Collecting the data set**

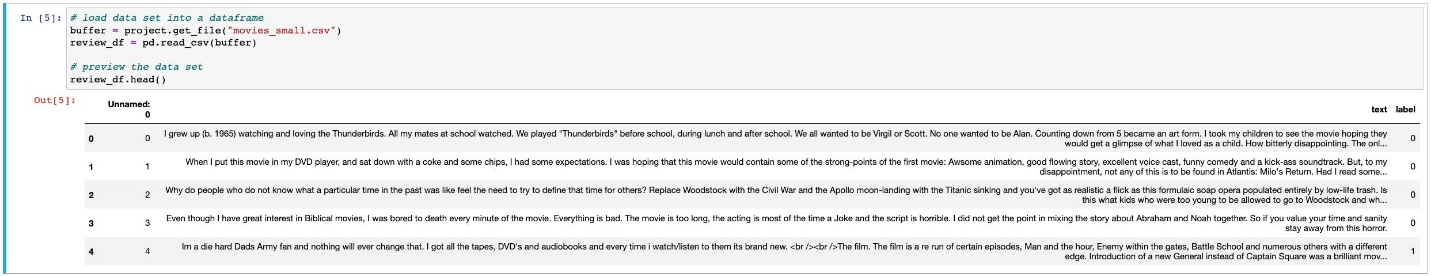
**1.Download the IMDB movie reviews data set . This data set from Kaggle has been down-sampled and saved for you to reduce the model training time.**

**2.Upload the data set to your Watson Studio project by going to the Assets tab and then dropping the data files, as shown in the following figure.**

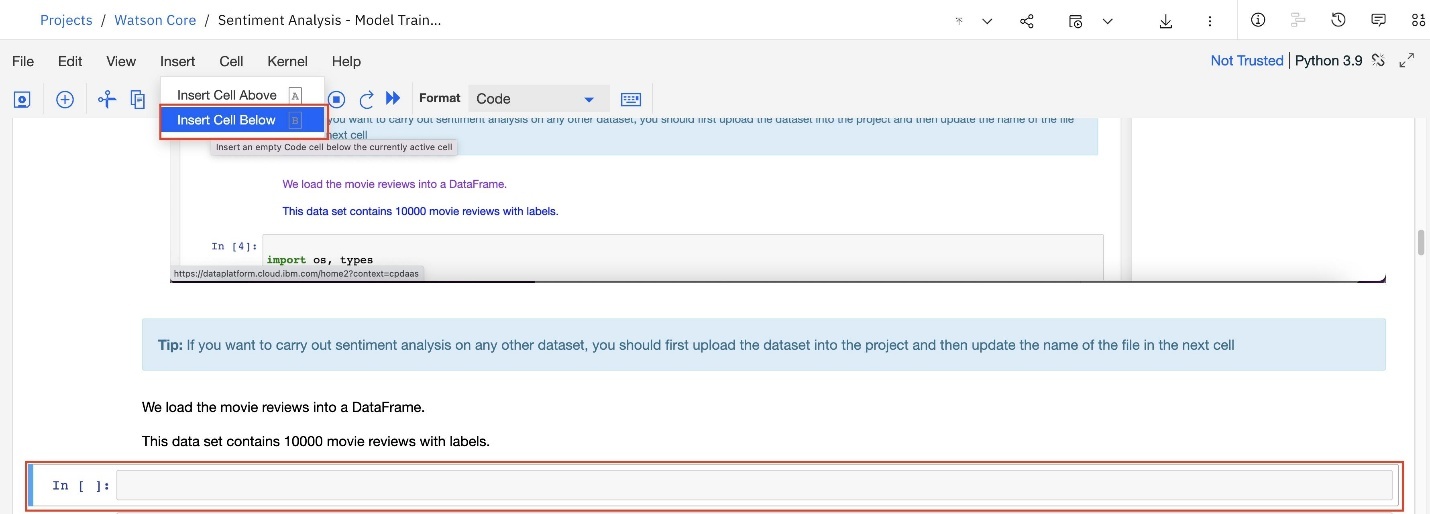
****

After you add the data set to the project, you might have to reload the notebook. You have two options of accessing the data set from the Jupyter Notebook depending on the level of access that you have.

1. If you are a project administrator, then:
   1. You can just insert the project token as shown in the following image.
   2. After inserting the project token, you can continue executing all of the cells in the notebook. This cell in particular loads your data set in the notebook.

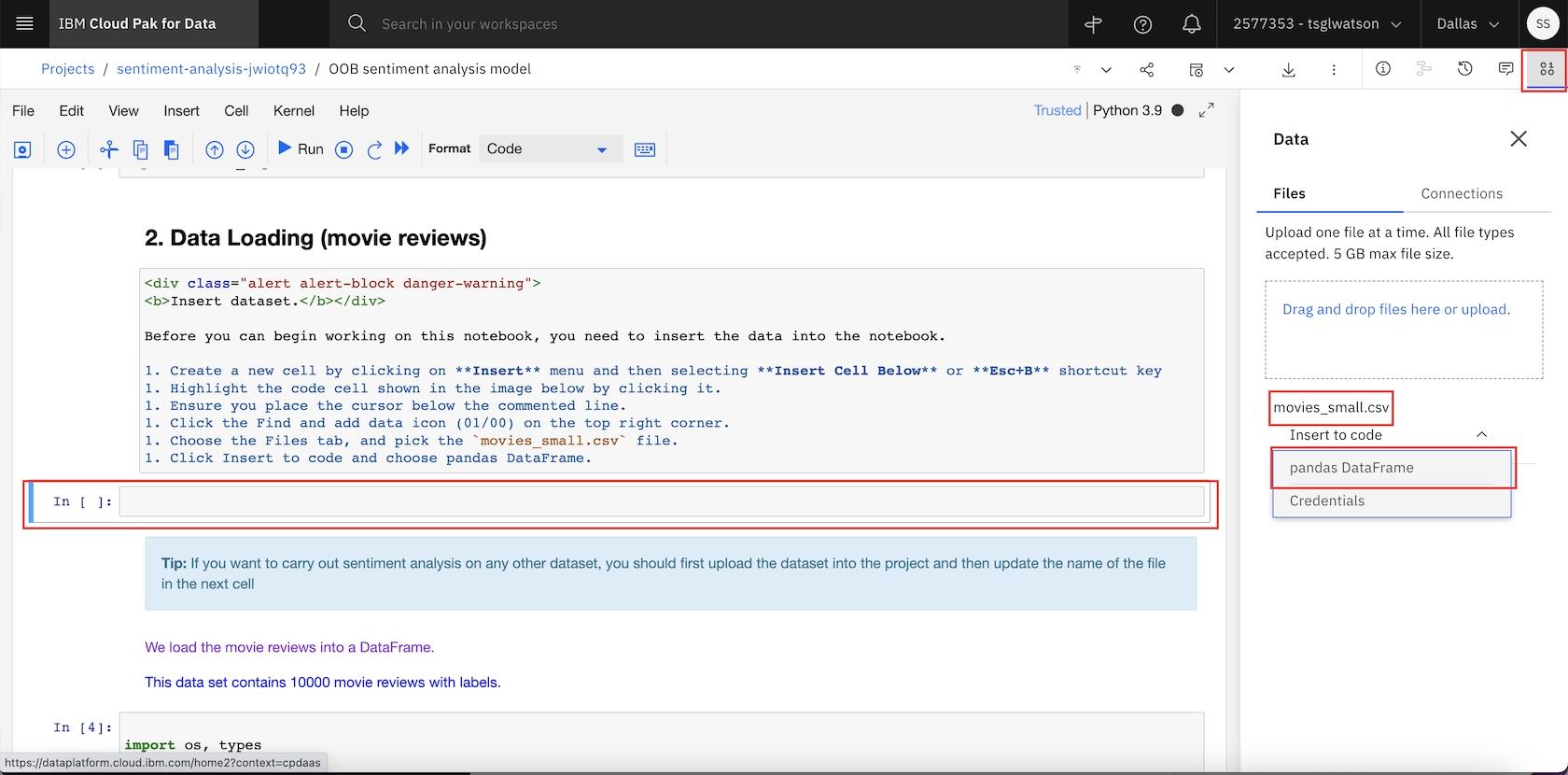
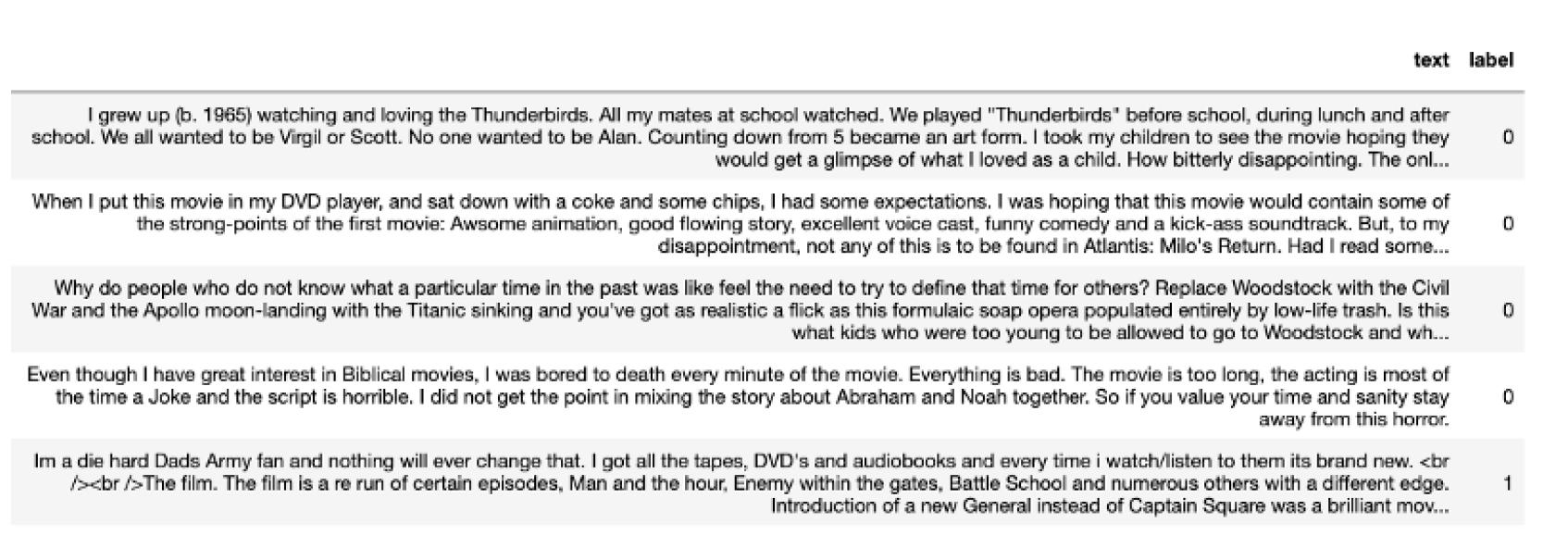


1. If you are not a Watson Studio project administrator, then you cannot create a project token.
   1. Create a new cell under Step 2 – Data Loading by clicking the Insert menu, and then selecting Insert Cell Below or the Esc+B keyboard shortcut. Highlight the code cell that is shown in the following image by clicking it.



* 1. Ensure you place the cursor below the commented line. Click the Find and add data icon (01/00) at the upper-right. Choose the Files tab, and pick the movies\_small.csv file. Click Insert to code and choose pandas DataFrame. Rename the DataFrame from df\_data\_1 to review\_df.

After you’ve added the data set to the project, you can access it from the Jupyter Notebook and read the .csv file into a pandas DataFrame.

l

Step 2. Data processing and exploratory data analysis

Step 2.1. Extract frequently mentioned movie aspects

In this step, you use the IBM Watson NLP pretrained syntax model to extract the most frequently mentioned aspects (nouns) in the movie reviews for each movie to understand what the audiences are talking about in their reviews.

Load the IBM Watson NLP syntax model.

Syntax\_model = watson\_nlp.load(watson\_nlp.download(‘syntax\_izumo\_en\_stock’))

The syntax model performs parts-of-speech tagging (for nouns, pronouns, verbs, and so on) so that you can filter out all of the nouns from the reviews. Instead of retrieving the nouns as they occur in the review, the lemmatized version of the nouns (for example, companies -> company) is stored. This lets you create more accurate statistics over all noun occurrences.

Use the helper functions to identify the lemmatized form of nouns in the review text.

Def extract\_nouns(review\_text):

# converting text into lower case before processing

Review\_text =review\_text.lower()

Syntax\_result = syntax\_model.run(review\_text, parsers=(‘lemma’, ‘part\_of\_speech’))

# filter nouns

Nouns = [token[‘lemma’] for token in syntax\_result.to\_dict()[‘tokens’] if ((token[‘part\_of\_speech’] == ‘POS\_PROPN’ or token[‘part\_of\_speech’] == ‘POS\_NOUN’) and len(token[‘lemma’]) > 2)]

# remove stopwords

Nouns = [noun for noun in nouns if noun.upper() not in stopwords]

Return list(nouns)

Extract the lemmatized nouns and show them with the review text in which they occurred.

Noun\_df = create\_noun\_dataframe(review\_df)

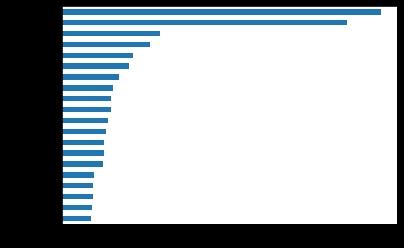
Sentiment\_noun\_df = review\_df[[‘text’, ‘label’]].merge(noun\_df, how=’left’, left\_index=True, right\_index=True)

Sentiment\_noun\_df.head()

Use the explode function to transform the noun list to separate rows for each noun. That way, you can count the occurrences for each noun in a subsequent step.

Exp\_nouns = sentiment\_noun\_df.explode(‘Nouns’)

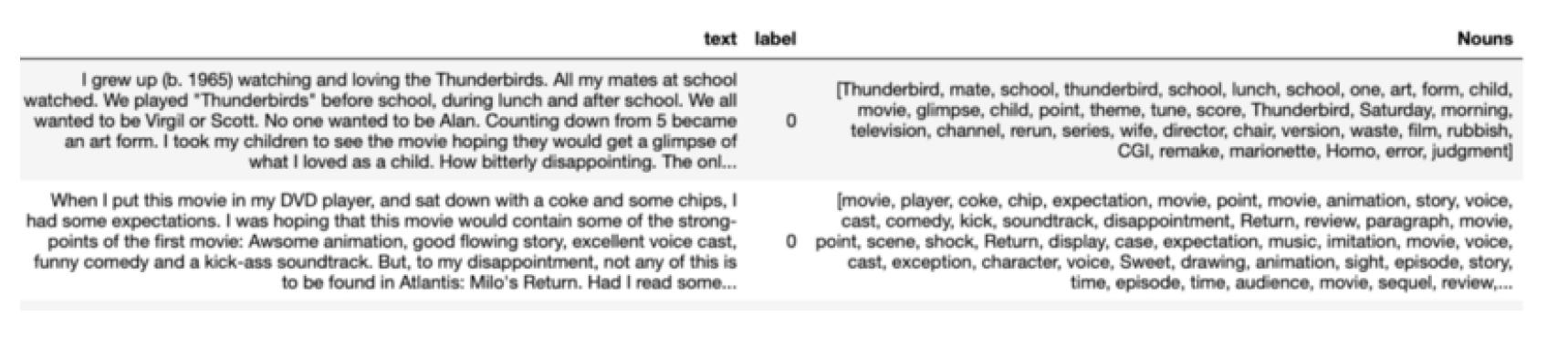
Plot the most frequent nouns as a bar chart.

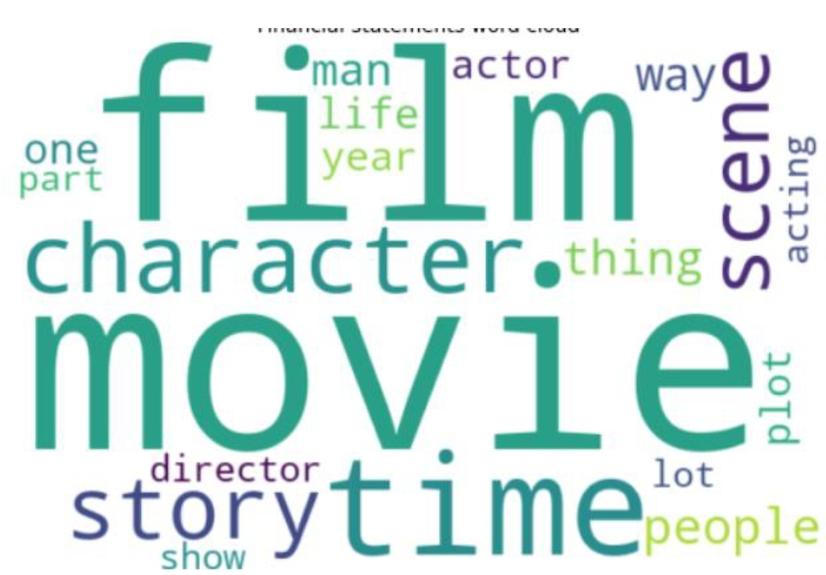


Create a word cloud for the most frequent nouns and show them.



This block extracts nouns from the movie reviews. The most frequently used nouns are typical aspects of a movie that review authors talk about.





Step 3. Model building

Step 3.1. Extract document and sentence sentiment

Are reviewers talking positively or negatively about the movies? Sentiment can be extracted for the complete review and for individual sentences. The sentiment extraction helpers can extract both sentiment levels.

Load the sentiment-aggregated\_cnn-workflow\_en\_stock sentiment model for English.

Sentiment\_model = watson\_nlp.load(watson\_nlp.download(‘sentiment-aggregated\_cnn-workflow\_en\_stock’))

Extract the overall sentiment of the review and the sentiment for each sentence.

Def extract\_sentiment(review\_text):

# run the syntax model

# converting review text into lower case

Review\_text = review\_text.lower()

Syntax\_result = syntax\_model.run(review\_text, parsers=(‘token’, ‘lemma’, ‘part\_of\_speech’))

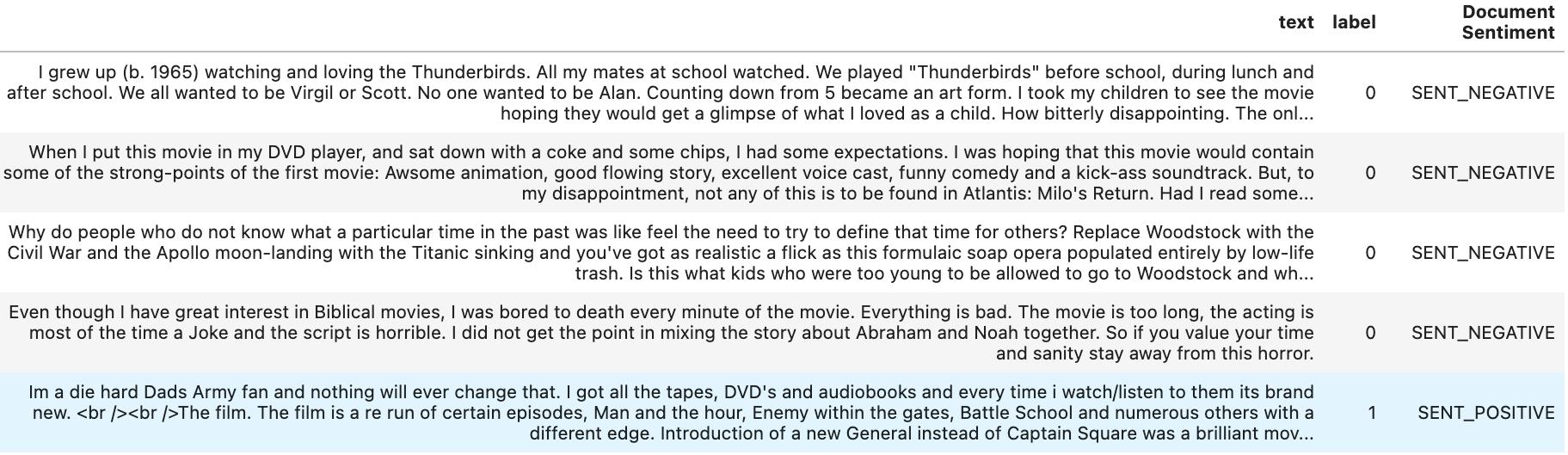
# run the sentiment model on the result of the syntax analysis

Sentiment\_result = sentiment\_model.run(syntax\_result, sentence\_sentiment=True)

Document\_sentiment = sentiment\_result.to\_dict()[‘label’]

Sentence\_sentiment = [(sm[‘span’][‘text’], sm[‘label’]) for sm in sentiment\_result.to\_dict()[‘sentiment\_mentions’]]

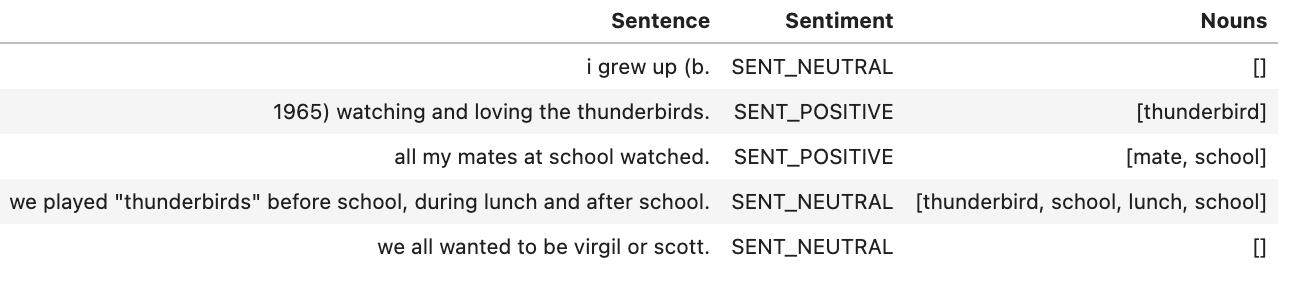
Return (document\_sentiment, sentence

Extract the sentiment and display it with the review text.

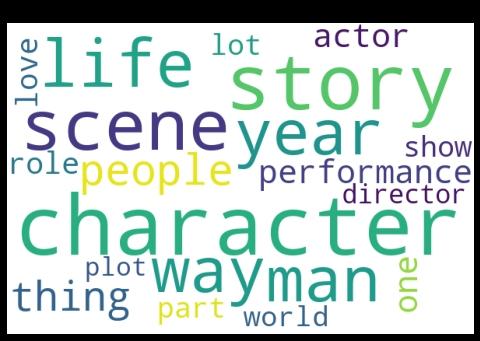
Step 3.2. Identify nouns that drive sentiment

You now identify the most frequently used nouns in sentences with positive or negative sentiment.

Extract nouns from the sentences/sentiment DataFrame.

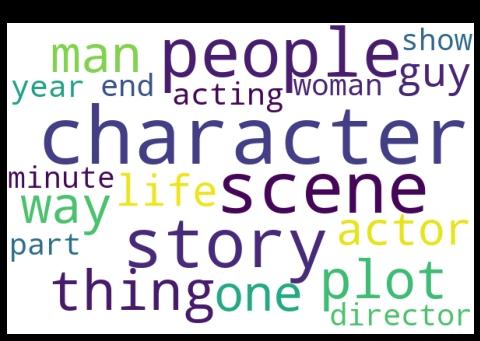


Show the most frequent nouns in positive sentences.



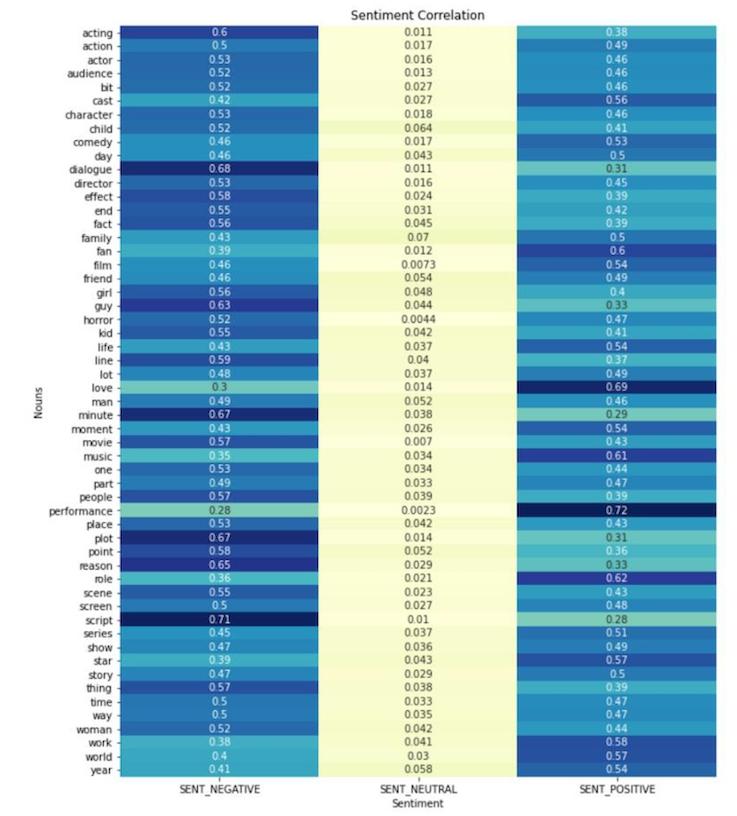
In this case, the word cloud shows that the most positive sentiments come from the reviews with words like film, movie, time, story, and character. So, the audience must love the film or movie when the story and character were good, and these were mentioned in the reviews.

Show the most frequent nouns in negative sentences.



In this case, the word cloud shows that the most negative sentiments come from reviews with words like time, character, scene, and director. So, the audience might not have liked movies with poor direction, character, and scenes in the movie.

Show the nouns that “drive” sentiment for each review. Create a cross tab between nouns and the resulting sentence sentiment and correlate them. The darker the cell, the more often a noun occurs in a sentence of a certain polarity.



You can see from the previous image that script is the most contributing factor to negative sentiments while performance and love are the most contributing factors to positive sentiments.

Step 3.3. Aspect-oriented sentiment analysis

Let’s see what the sentiment for these aspects (script, performance, and love) looks like. For this, use the Watson NLP targeted sentiment model to extract the sentiment that is specific to the most frequently used nouns in the movie reviews.

Load the syntax model for English and the targeted BERT sentiment model.

Targeted\_sentiment\_model = watson\_nlp.load(watson\_nlp.download(‘sentiment-targeted\_bert\_multi\_stock’))

Use a helper method to extract the sentiment for each target and the sentences containing the target as evidence.

Def extract\_targeted\_sentiment(review\_text, target\_words):

# run the syntax model

Syntax\_result = syntax\_model.run(review\_text, parsers=(‘token’, ‘lemma’, ‘part\_of\_speech’))

# extract the spans for the target words

Targets = watson\_nlp.toolkit.get\_target\_spans(review\_text, target\_words)

# run the sentiment model on the result of syntax and the target words

Sentiment\_result = targeted\_sentiment\_model.run(syntax\_result, targets)

# iterate over all target aspects

Target\_sentiments = []

For idx, val in enumerate(target\_words):

Sentiment\_prediction = sentiment\_result.to\_dict()[‘sentiment\_predictions’][idx]

Sentiment\_mentions = None

If sentiment\_prediction[‘sentiment\_mentions’] is not None:

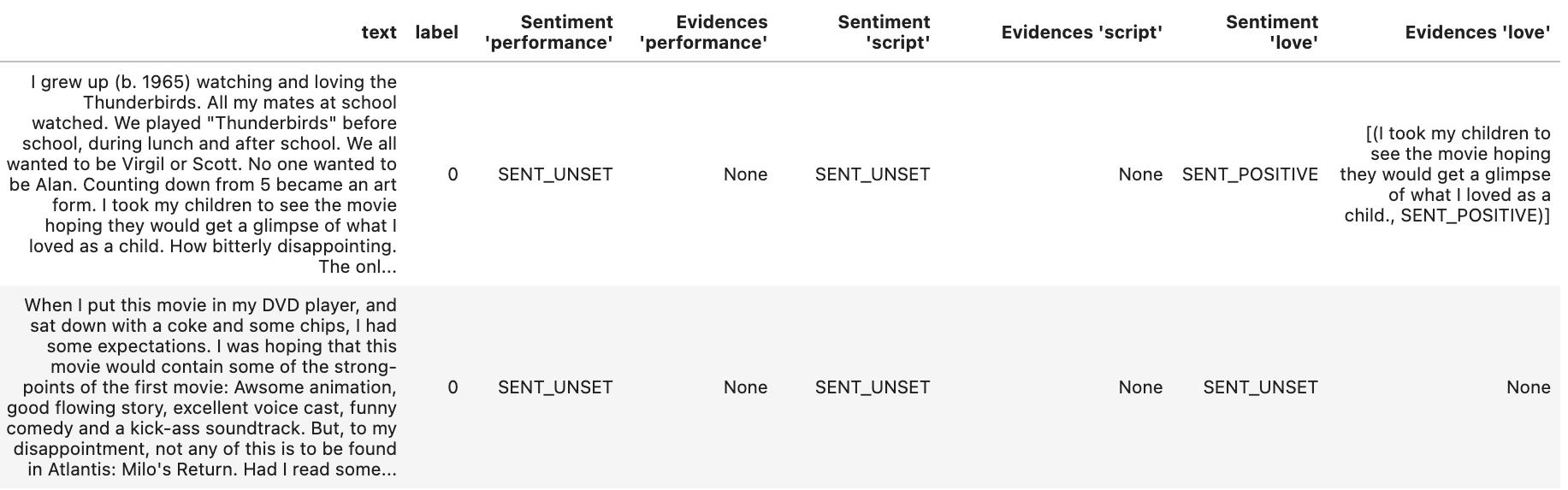
Sentiment\_mentions = [(sm[‘span’][‘text’], sm[‘label’]) for sm in sentiment\_prediction[‘sentiment\_mentions’]]

Target\_sentiments.append(sentiment\_prediction[‘label’])

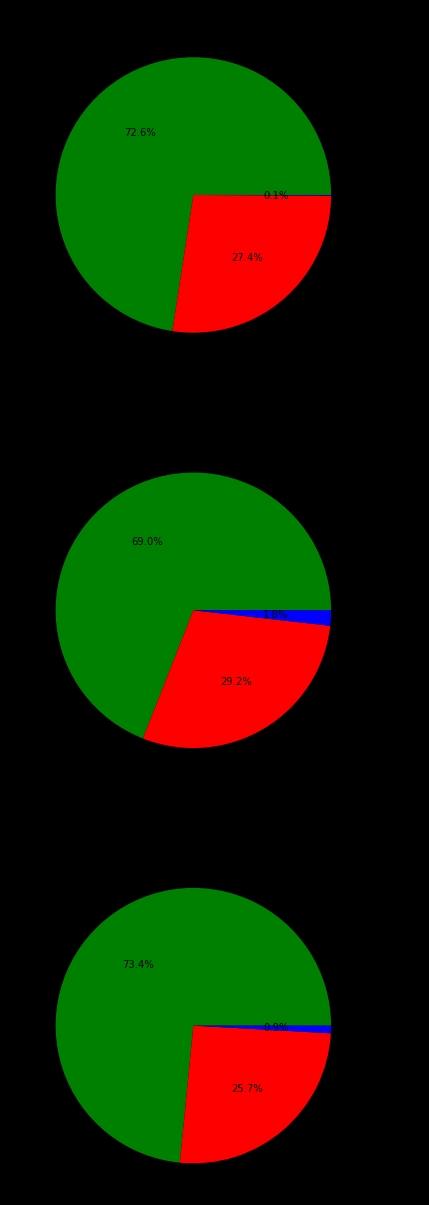
Target\_sentiments.append(sentiment\_mentions)

Return target\_sentiments

Extract the sentiment specifically for script, performance, and love. Note that this cell runs for several minutes.



Display the sentiment for selected aspects.



Step 4. Model evaluation

Apply data processing on the test data to make it compatible with the evaluate method in the Watson NLP library.

Def input\_data\_prep(df):

Df[‘weight’] = 1

Df.rename(columns={‘label’: ‘labels’}, inplace=True)

Df = df[[‘text’, ‘weight’, ‘labels’]]

Df[‘labels’] = df[‘labels’].replace({0: ‘negative’, 1: ‘positive’})

Df[‘labels’] = df[‘labels’].apply(convertToList)

Display(df.head(5))

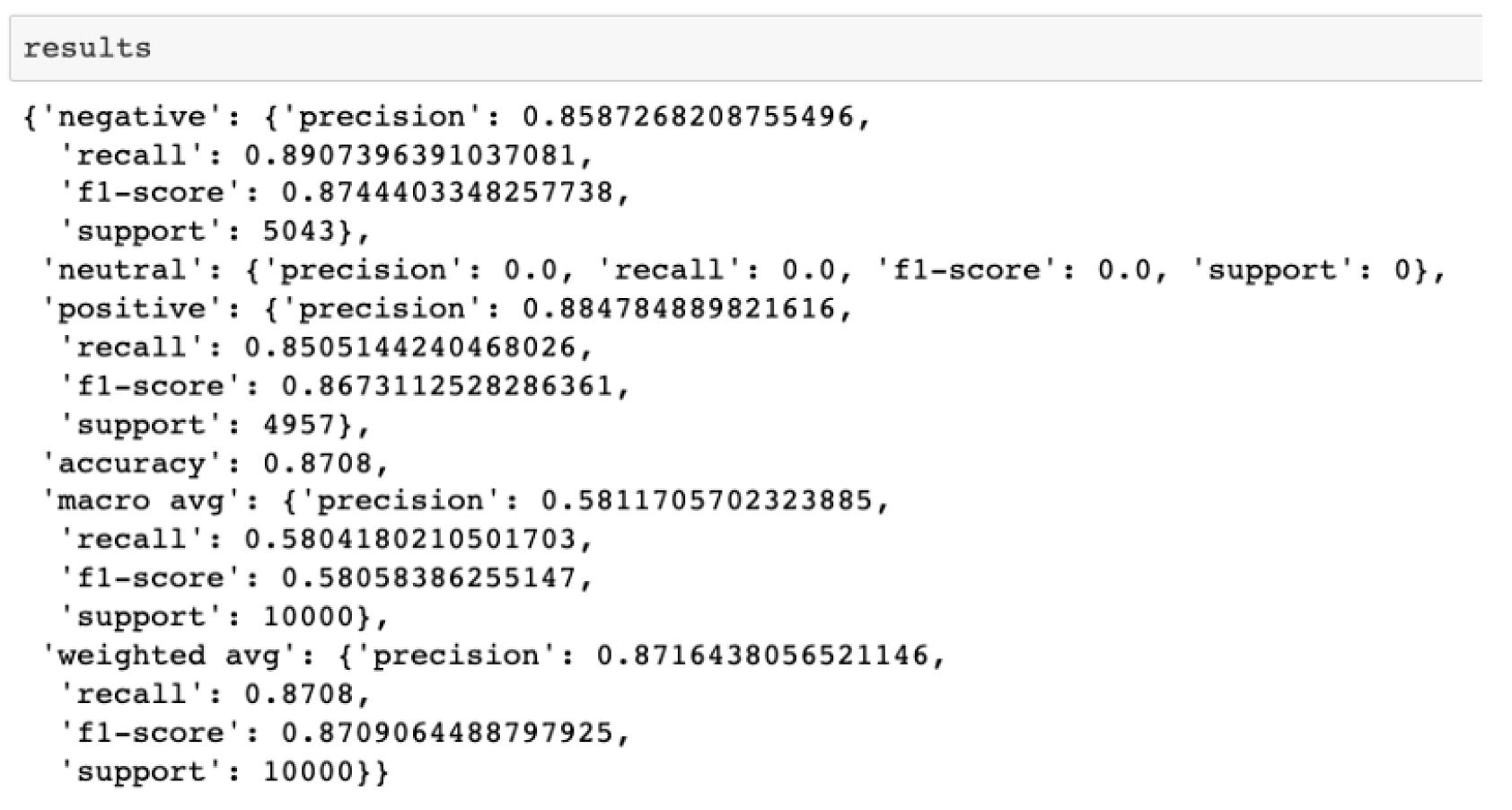
Return df

Evaluate the model by passing the test data and using the evaluate\_quality method from the Watson NLP library.

Pre\_eval\_func = lambda x: syntax\_model.run\_batch(x)

Sentiment\_model.find\_label\_func

Results = sentiment\_model.evaluate\_quality



As you can see, the overall accuracy, precision and recall values are 0.87 each. This has been achieved by evaluating a pretrained model without training it on the IMDB Movie reviews data set

***Git hub link :***