**Spring School: Data Science Methods within Geo-Informatics**

**Final Project: Evaluating the Performance of a Location Extractor Model using the f-beta Score.**

**Introduction**

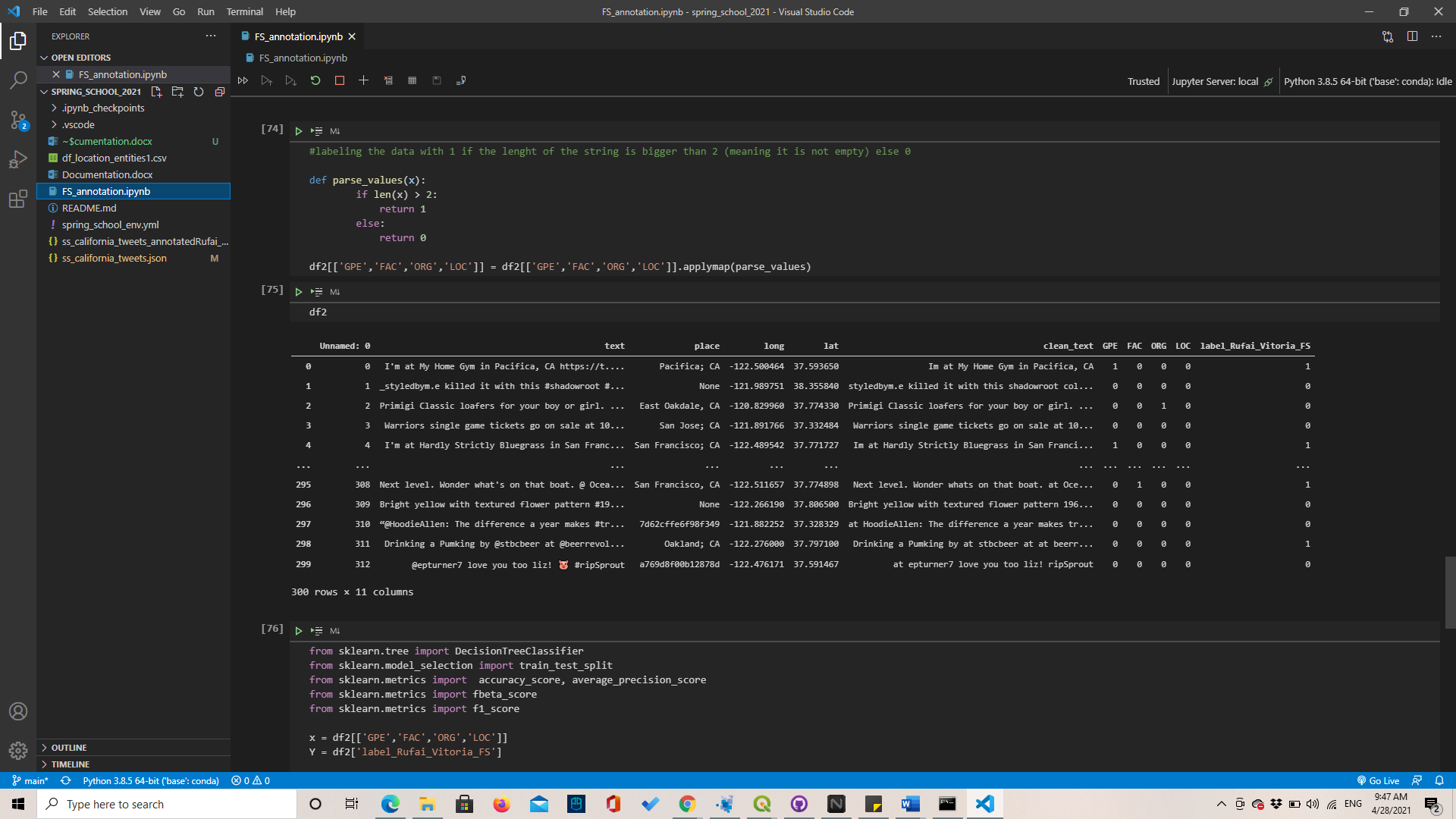
These days, the availability of large data streams –structured and unstructured – gives us the opportunity to extract information for different use cases that did not exists until now. This information extraction process is useful in multiple domains and fosters innovations, support policy and decision making and serves as the starting point of many research interests. Within the geoinformatics domain, central to these information extraction techniques, is the extraction of location information in form of latitude, longitude, street names, states, county, and other information entities that relates to location. An important source of location data these days is social media and search engine platforms. For instance, Google processes over 3.5 billion search queries everyday and at least 500 million tweets are made every day.

In most cases, through accessible social media APIs and text databases, thousands of text information can be access and analysed. At the heart of this information extraction process is Natural Language Processing, which serves multiple research interests. In this case, we utilized the Spacy library developed by Explosion to extract the location information from a set of social media posts.

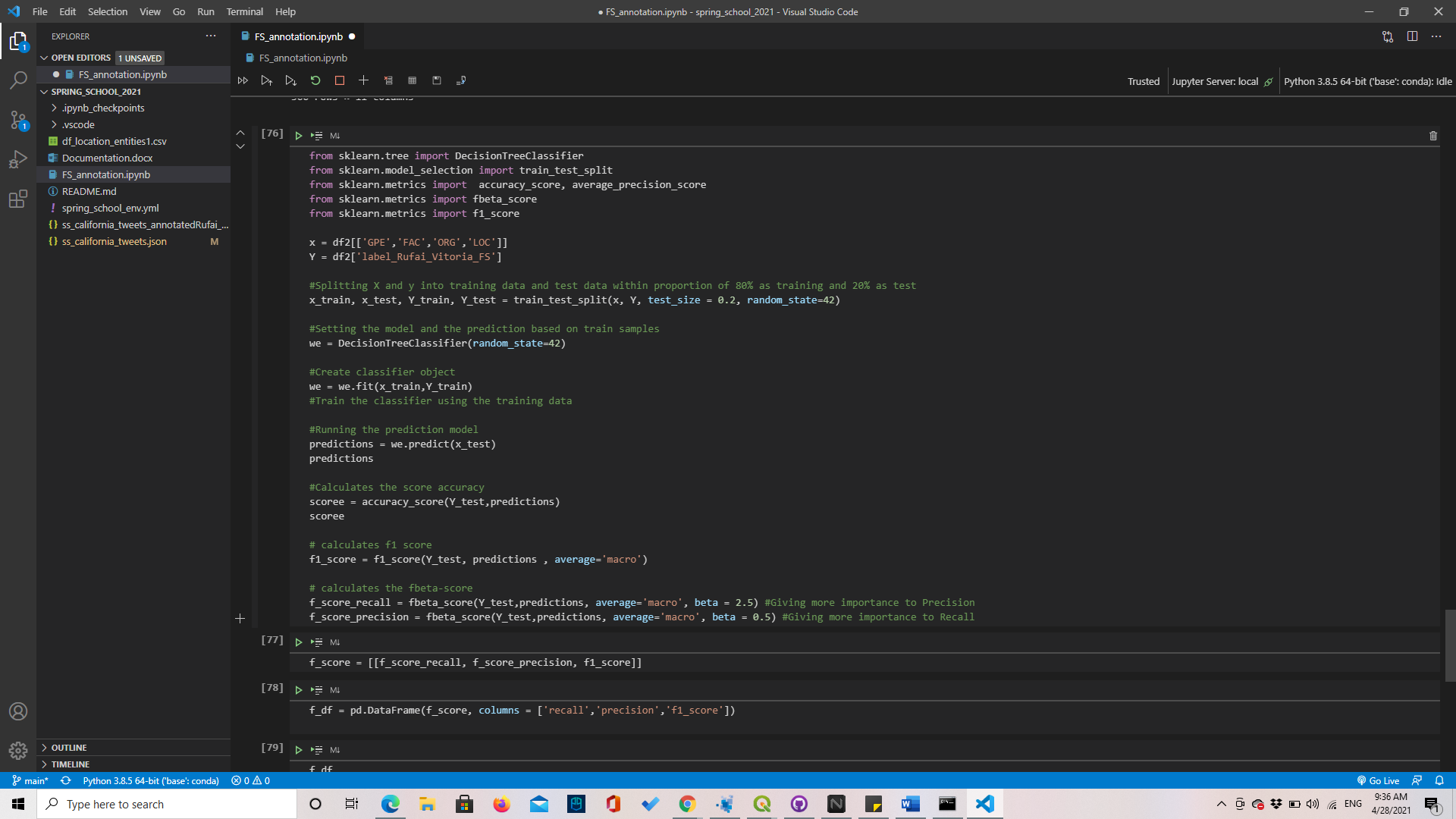
**Methods**

After applying the Named Entity Recognition algorithm available in SpaCy, the newly generated dataset with location information was annotated using a binary class of 1 (contains location entity) and 0 (no location entity). This was based on the guideline provided by the Spring School annotation guideline. To fast track the analysis, the first 300 sets of the social media posts were annotated and coded. The annotated data was store in a JSON file (*ss\_california\_tweets\_annotatedRufai\_Vitoria\_FS*) accessible in the files folder.

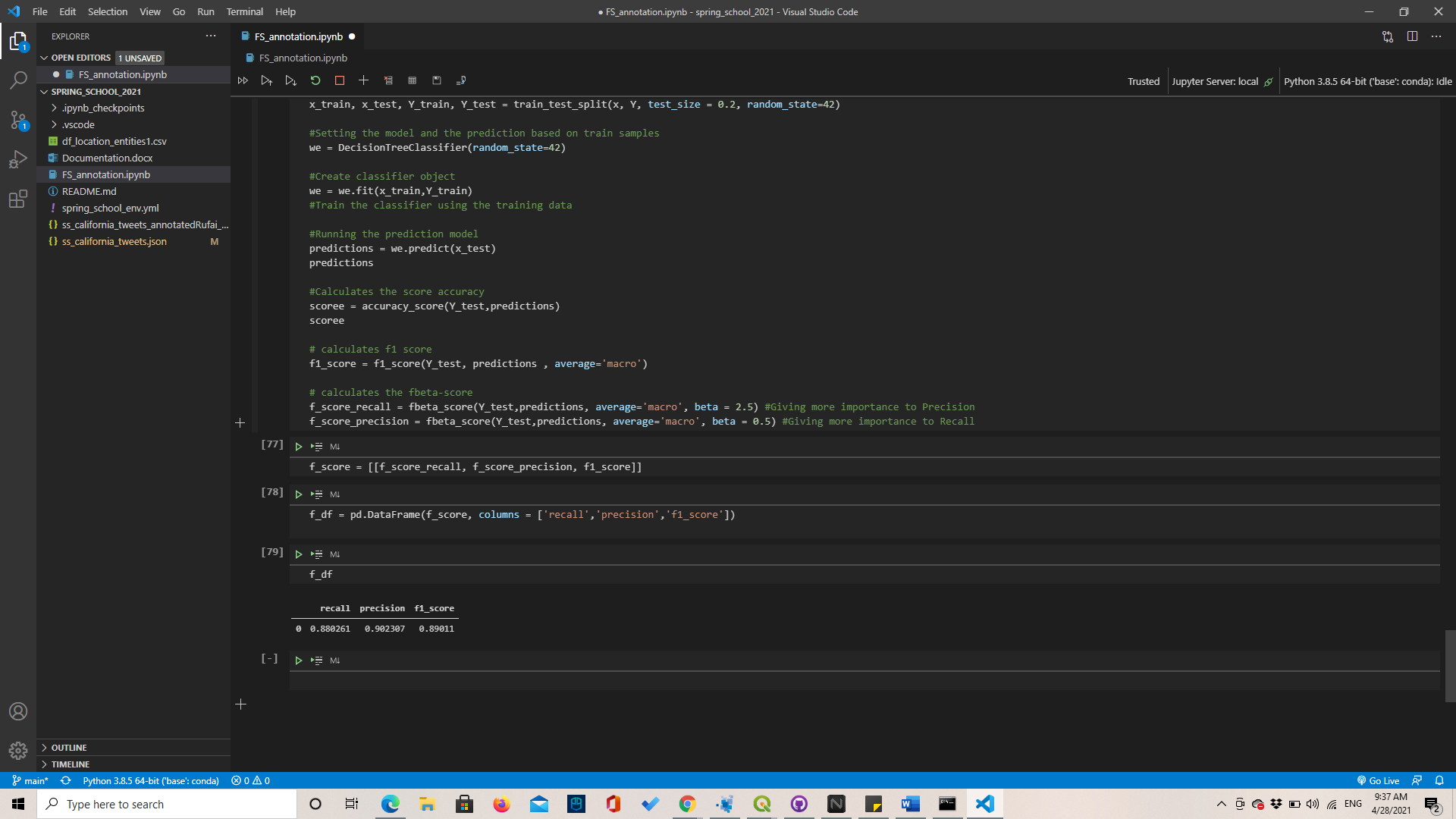
In order to develop a machine learning model that could be used to predict the location of other text information, we designed a simple Decision tree classifier. This was done after the location entities fields have been vectorized in a binary form as well to match the label field, see the figure below.



The dataset was split into training (80%) and test sets (20%) and the model was trained on the annotated data, and a prediction was carried out on the test sets. The random state was set to 42, to minimize variation in model output at each run. The code cell below shows how these algorithms was called from the SK-Learn API and applied to our use case.



Afterwards, the evaluation metrics (f1 scores and score) were computed. We wanted to understand our well the model performed in identifying true negatives and true positives. To do this, we need an evaluation metric that can suitably identify the different weightings of recall and precision. In this case, score (also available in SK-learn API) is the most robust metric to use. The harmonic means of recall and precision -- f1-Score – provides a weighted balance between these precision and recall. To best evaluate the model performance, the extracted locations were coded as 1 and 0, in the spacy extraction of locations entities [FAC, GPE, ORG and LOC] to fit with the annotated sets. See the computed value below:



|  |  |  |
| --- | --- | --- |
| Recall | Precision | f1 Score |
| 0.880261 | 0.902307 | 0.89011 |

The table above shows a summary of the output of evaluation metric with the computation of the differing focus on recall, precision and just the weighted mean (f1 score).

The score, which is the harmonic mean of both recall and precision, shows that without considering any class imbalances the model performed at approximately 89% accuracy. Meaning that, 89% of the time the model will return a True Positive and 11% of false positive. The other two score which prioritize one of recall or precision shows the evaluation based on the class imbalances for the application context. In this case, we have not particular application context in mind, since the sets of social media posts used for this analysis were not focused on a particular topic but rather covers a range of interests. The recall with a beta value of 2.5 shows that recall is 2.5 times as important as precision. In this case, the computation returned a lower value of 0.880261 compared to the score, meaning that the model has good precision but poor recall. This is corroborated by the value of which weighs precision more than recall and returned a value of 0.902307.

**Discussion and Considerations**

By and large, the cost of deciding which evaluation is considered robust depends on the application context. Hence, to reach a good conclusion on what score to assign to the beta parameter in the f-beta score, we must consider the relative cost of a false positive and false negative, especially when the class distributions are not balanced. Given an application context of locating endangered persons during a humanitarian crisis, where saving lives with limited resources is a priority, it will be better to have a higher recall than precision to save the most lives. In such case, we can easily tweak the f-beta score to evaluate for higher (give more weight to) recall over precision. Additionally, we should keep in mind that the computation of precision, recall and f-beta do not count the rate of true negatives. If this is important to a particular use case, then a different evaluation metric might do the job.

An additional question to keep in mind is **how to approach the co-extraction of noise with a correct location entity.**

**Lemmatization,**

**Stemming,**

**Conclusion**

**References**

What is F-Score? Thomas Wood (n,d) retrieved from https://deepai.org/machine-learning-glossary-and-terms/f-score