PredictingWaterQuality

Oguzhan SAHIN-Ozgur DOGAN-Dilara SAHAN

January 2021

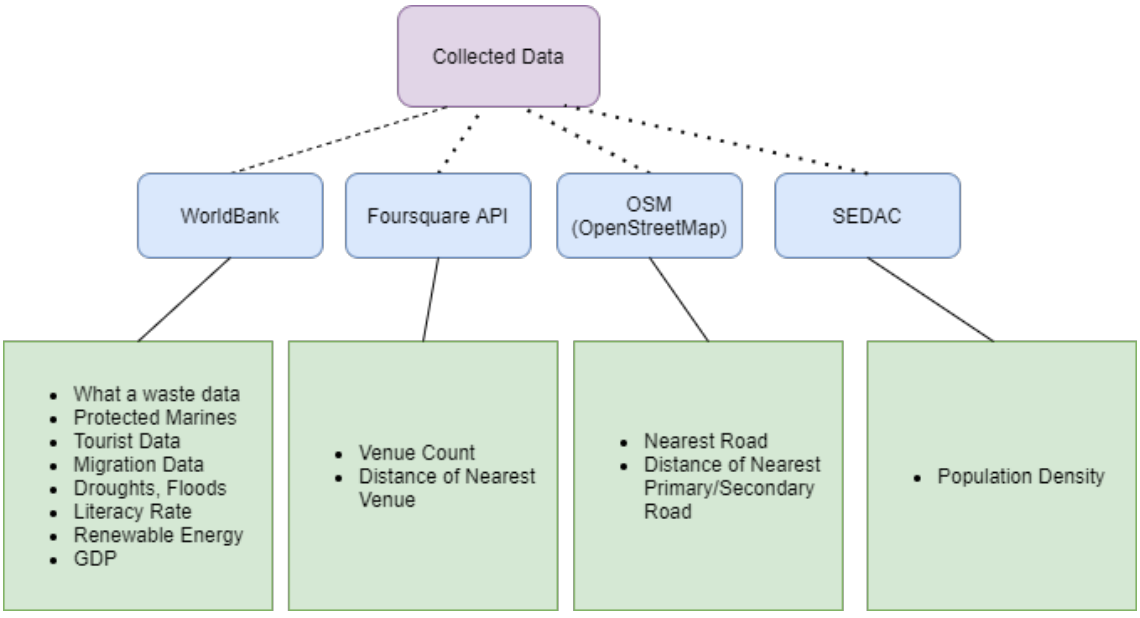
1 Abstract

Humanity have some fundamental needs for existing and the most important of these is water. However, some people continue to pollute lakes, seas and oceans. Water pollution is mostly a result of oil spills and industrial wastes. On the other hand, studies are carried out in many areas to prevent water pollution. When we heard about the presence of algae that clean water and increase the water quality , we decided to focus on this issue. In this project, we aimed to create a model that predicts which waters needed this algae.

2 Method

2.1 Data Collection

After identifying our problem, first we focused on data collection, which is the most important part of our ML pipeline. For this, we first looked for data that we can work on that shows water quality and decided to use the data europa water pollution data set. Since most features will not be effective for the model, we dropped most of them and decided to add new features. Accordingly, we decided to use WorldBank, SEDAC, Foursquare and OSM (OpenStreetMap) data. The data we add is as follows.



2.2 Data Preparation

Since the data we collect is generally country-based, we first obtained country information from the coordinate information on our main water pollution dataset. We have joined data according to this country information. Apart

from this ready-to-use data, we decided to use the FourSquare API to detect venues close to the given coordinates, and the OSM API to detect nearby main roads, secondary roads and motor paths. But did not use the OSMnx package due to the lack of computational power.

2.3 Data Preprocessing

We combined the different data we collected in a table and made our last studies / tests on this dataset. First of all, we applied normalization because we wanted to get rid of the effect of outliers in the data set. Some outlier values were removed manually from the data set. Then, because categorical data poses a problem, we converted the categorical data into numeric classes with the Label encoder, and we dropped the nan values in the data because they were almost non-existent.

2.4 Data Exploration

After finalizing the data, we analyzed and visualized the data in order to understand the connections and patterns within the data. Here is some analysis:

2.4.1 Analysis Data

If we need to analyze the data quickly, the final version of the data contains 20k tuples and 29 features. If we need to look at the data types of these 29 attributes, we have 8 categorical and 21 numeric data types. Among these 29 features, the column we will use as the target is the "resultMeanValue" column that contains a continuous data. As

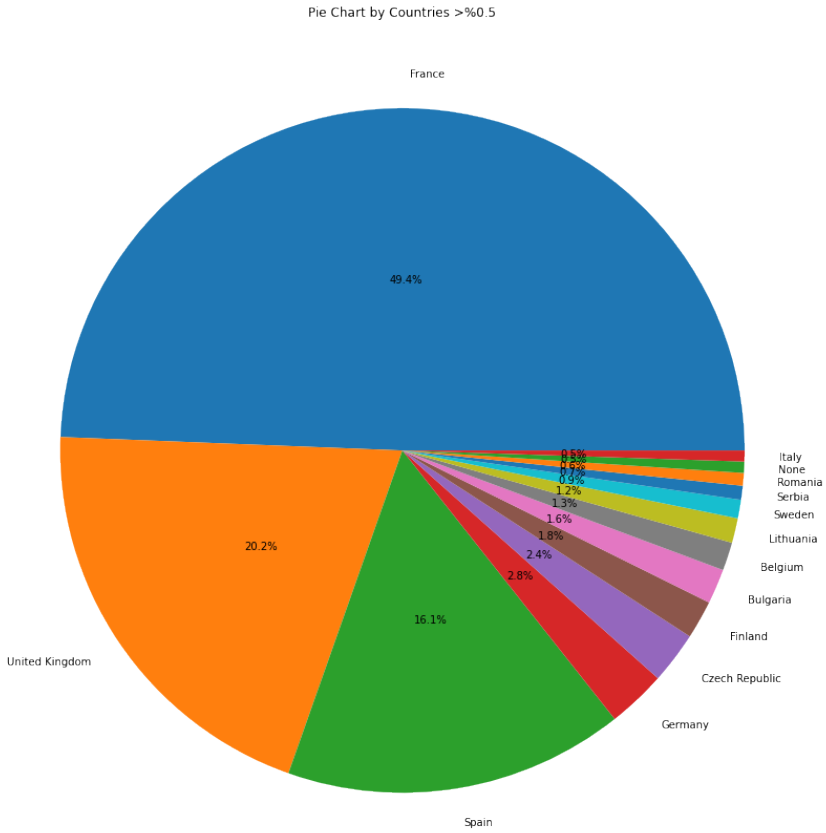
```
data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20000 entries, 0 to 19999
Data columns (total 29 columns):
 #   Column                                  Non-Null Count  Dtype
---  ------                                  --------------  -----
 0   parameterWaterBodyCategory              20000 non-null  object
 1   observedPropertyDeterminandCode         20000 non-null  object
 2   procedureAnalysedFraction               20000 non-null  object
 3   procedureAnalysedMedia                  20000 non-null  object
 4   resultUom                               20000 non-null  object
 5   phenomenonTimeReferenceYear             20000 non-null  int64
 6   parameterSamplingPeriod                 20000 non-null  object
 7   resultMeanValue                         20000 non-null  float64
 8   waterBodyIdentifier                     20000 non-null  object
 9   Country                                 20000 non-null  object
 10  PopulationDensity                       19893 non-null  float64
 11  TerraMarineProtected_2016_2018          19893 non-null  float64
 12  TouristMean_1990_2020                   19893 non-null  float64
 13  VenueCount                              20000 non-null  float64
 14  netMigration_2011_2018                  19893 non-null  float64
 15  droughts_floods_temperature             19893 non-null  float64
 16  literacyRate_2010_2018                  19893 non-null  float64
 17  combustibleRenewables_2009_2014         19893 non-null  float64
 18  gdp                                     19893 non-null  float64
 19  composition_food_organic_waste_percent  19893 non-null  float64
 20  composition_glass_percent               19893 non-null  float64
 21  composition_metal_percent               19893 non-null  float64
 22  composition_other_percent               19893 non-null  float64
 23  composition_paper_cardboard_percent     19893 non-null  float64
 24  composition_plastic_percent             19893 non-null  float64
 25  composition_rubber_leather_percent      19893 non-null  float64
 26  composition_wood_percent                19893 non-null  float64
 27  composition_yard_garden_green_waste_percent  19893 non-null  float64
 28  waste_treatment_recycling_percent       19893 non-null  float64
dtypes: float64(20), int64(1), object(8)
memory usage: 4.4+ MB
```

stated in the data preprocessing section, we decided to delete the rows with these values since there are almost no NaN values in the data set. If there were more we could have preferred to fill in these NaN values, but their small number pushed us to this method.

2.4.2 Visualization Data

We made visualizations using a Seaborn, Matplotlib and Folium libraries to better understand the data. Our first visualization was on the distribution of water samples taken across countries. Since the raw version of the data set contains more than 2 million data, we had to make certain clippings. Most of the last remaining data are in the regions of France, England and Spain.You can access all the visualizations we have made on the EDA notebook in the github repository. In addition, we visualized these data on a map according to their coordinates with the help



of the Folium library to learn how the samples are distributed on the map.

2.5 Predictive Models

Our main problem was a regression problem because of the data we chose. "resulltMeanValue", which is our target feature, contains continuous values. However, we handled the problem both as a classification and regression problem.

2.5.1 Classical Classification Algorithms

For the classification problem, we first chose to use classical ML algorithms. The algorithms we have chosen for this are XGBoost, KNN, SVC, Random Forest, Ada Boost and Naive Bayes. We created different models and trained and predicted with the same data. We made our target attribute categorical by specifying a threshold value. The models gave the following results.

Model	Accuracy	ROC/AUC Score
XGBoost	0.9892	0.9889
KNN	0.9877	0.9171
SVC	0.9867	0.9346
Random Forest	0.9906	0.9357
AdaBoost	0.9884	0.9836
Naive Bayes	0.6575	0.8351

2.5.2 Regression

When we wanted to solve the problem with a regression algorithm, we chose the XGBoost regressor, which is the favorite algorithm of Kaggle competitions. With the XGBoost regressor, we got the following results according to certain metrics.

Metric	XGBoost Regressor Score
MSE	26658.278980488074
RMSE	163.273633222666443
R^2 Score	0.14316101172408313

2.5.3 Classification by Features

In line with the decision we made as a result of our interview in this section, we run the model by adding the features one by one to see how effective the features are. But the features did not affect the accuracy much.

3 Results

In line with our studies, we found that it was difficult to collect data from the very beginning and work with them. We could not achieve the results we wanted in models resulting from our data selection. Even though we get high scores, we think it's overfit. In the following days, we will do tests with unseen data and we will make the models even better.

4 Conclusion

As a conclusion, it was a fun project for us. We used APIs that we did not use, did long-term research for data, and created a data set from scratch. This added a lot to us. We hope this project can go further.