Zipcode Plugin Documentation

Overview

The Zipcode Plugin helps preprocess and analyze a dataset column to calculate a confidence score for how likely each value in the column is a valid US zip code. The plugin has three main parameters for calculating the score: the column name, the format of the value passed, and an external API checker.

Dependencies/ Pre-requisites

Regular Expression module - **Re**

statistics module

os module- to use os functionality to get environment variables

opencage - a package that provides the OpenCage API service. To get access to this API:

1) Open the command prompt/ terminal window
2) Type “pip install opencage”
3) Create an Opencage account
4) Get the API key associated with your account: Geocoding API → Quickstart
5) Copy the key
6) Create a .env file in your project directory
7) Type the code: `OPENCAGE_API_KEY= “paste your API key here”,` and save the file

Function Definitions

get_confidence_score()

The get_confidence_score() function takes in the column name as a string and a sample of the dataset column as a list of strings.

The function checks two conditions for the column name:

1) Condition one: If the column name contains the substring "zipcode" or "postal code”. A boolean variable “colcheck_1” acts as a checker and is set to True if the column name matches the condition.
2) Condition two: If the column name contains the substring "zip" or "postal”. A boolean variable “colcheck_2” acts as a checker and is set to True if the column name matches the condition.

Note: if colcheck_2 is True colcheck_1 will automatically be True
Variable colcheck” is set to Yes/Maybe/No depending on the values of colcheck_1 and colchec_2
If both are True the value is set to “Yes”
If only colcheck_2 is true it is set to “Maybe”(case of ambiguity)- as “zip” could also mean a column with zip files.
If Both are False, the value is set to “No”
Depending on the value of colcheck we would adjust the confidence score accordingly.

The next step is to preprocess the data: the list of values is sent to functions that remove all the null values and leading/trailing spaces from the data.

The function then initializes a list of scores to keep track of all the confidence score values for the column. Each value is passed through various conditions and the scores list is appended accordingly. Finally, the function returns the median of all the scores, which is the final confidence score of the entire column.

Variables used:
col_name- column name (String)
col_values- column values (list of Strings)
colcheck_1- condition one for column check (Boolean)
colcheck_2- condition two for column check (Boolean)
colcheck- the final result of the column name depending on the conditions (String)
scores- confidence score for each value (list of Integers)
format_check- regular expression format checker (Boolean)
Api_check-checks through the API (Boolean)
Input- col_name, col_values
Output- median value of scores (float)

remove_null()
This function removes all the null and missing values from the data. Values like NA, null, and Nan are removed to avoid skewness in the results.

remove_spaces()
This function removes all leading and trailing spaces (if any) from the values. This ensures that the data is properly formatted and consistent, making it easier to work with and reducing the chances of errors in the code.

get_format()
This function checks the format of the value passed to it. Valid US zip codes are five-digit (55555) or 5+4 digit codes (55555-5555). This function acts as a boolean checker that sets the value of format_check to True if the condition is satisfied.
Variables used:
c- each value from col_values
Input- c
Output- Boolean value

get_api_value()
This function checks the validity of a value passed to it through an external API. Opencage is a geolocating service that provides geocoding addresses. It uses a range of data sources to provide precise results. This API has an attribute "_type" that stores the type of the value being passed. For our function, we check if the result of "_type" is a postcode. The API key is accessed through the .env file created (details in the pre-requisites section). The value is then checked through the API and the result is stored as a JSON file. The _type attribute is extracted

from the JSON result and a boolean api_checker is set to True if the final result of the API is true.

Variables used:

c- each value from col_values
api_key- API key/password extracted from the .env file (String)
Geocoder- call the OpenCage API
api_json- stores the complete file of the result
api_result- stores the _type attribute
Input- c
Output- Boolean value

Conditional Reasoning:

This plugin checks through three parameters: column name(2 conditions), format, and the API. Unless there is any ambiguity in the results of column name and format, the value is not checked through an API (this is done to minimize the number of requests made to the API). The get_sconfidence_score() assigns the confidence scores according to the following conditions :

Condition	Score	Reasoning
format_check -True colcheck - Yes	100	Since the format matches the valid US zip code format and the column name matches both conditions, it is certain that the value is a zip code and it assigned a 100% score
format_check -True colcheck - Maybe	100	Since the format matches the valid US zip code format and the column name matches at least one condition, it would still mean that the value is a zip code, since format has a higher priority and it assigned a 100% score
format_check -True colcheck - No	Checks through API: 100 is True 0 if False	Since the format matches the valid US zip code format but the column name doesn't, it is a case of uncertainty and the value is checked through the API to make sure, the score is assigned to 100% or 0% depending on the result of the API
format_check -False colcheck - Yes	Checks through API: 100 is True	Since the format does not match the valid US zip code

	0 if False	format but the column name does, it is still possible that the value could be a spelling error/out of us zipcode (format has more priority) is checked through the API to make sure, the score is assigned to 100% or 0% depending on the result of the API
format_check -False colcheck - Maybe	Checks through API: 100 is True 0 if False	Since the format does not match the valid US zip code format and the column name is uncertain too, the value is checked through the API to make sure, the score is assigned to 100% or 0% depending on the result of the API
format_check -False colcheck - No	0	Since the format and the column name both does not match, it is certain that the value is not a valid zip code and the score is set to 0%

For version 1 of the plugin, only the API was used to calculate the final confidence score. Although the results were precise, two other parameters were added due to the following reasons:

- Performing tests by systematically trying every possible value through the API can significantly impact the efficiency in terms of time. This approach leads to longer processing times and increased computational demands which can reduce the overall performance. Therefore, alternative testing strategies were considered to help optimize the API's efficiency while ensuring comprehensive test coverage.
- Implementing alternative strategies also helped decrease the number of requests made to the API and reduced API usage costs.

This approach resulted in improved testing efficiency and cost saving for the organization.

Implications for the Future:

Although not needed, In the event of higher demand, a paid plan can be considered to meet increased user requirements.
OpenCahe Pricing details: https://opencagedata.com/pricing