Knowledge and Data - Final Project

Michelin Star Restaurants

Located in France

Group 9:
Sebastião Manuel Inácio Rosalino
Keerti Shetty
Andreas Jose Martinez
Vivi Jimenez

Index

1) Introduction

The present work focuses on the application of the evaluation criterion "*Michelin Stars*" to French restaurants and trying to understand what distinguishes the award-winning restaurants from the rest.

The *Michelin* Guides are a series of "guide books" that have been published by the French tire company *Michelin* since 1900. The Guide awards go up to three Michelin stars for excellence to a select few establishments. The acquisition or loss of a star or stars can have dramatic effects on the success of a restaurant. *Michelin* also publishes the Green Guides, a series of general guides to cities, regions, and countries. The *Michelin* Guide is the reference guide for hotels and restaurants. Printed with the utmost secrecy and with an unknown circulation, this guide is the most respected in the world and rewards the best restaurants, classifying them with stars that represent any chef's dream or nightmare. Winning a star in the guide means the rise of the restaurant and its chefs, while losing one of them can even lead to a tragedy like that of chef Bernard Loiseau, who committed suicide in February 2003, with 52 years old, desperate with rumor that his establishment would lose its "three star" rating in the Michelin Guide.

Primarily, *Michelin* stars represent the distinction of a restaurant in relation to others. Above all, Michelin stars represent excellent service and atmosphere as well as exceptional cuisine. As well as the presence of specialties and top-quality wine list. In summary, the presence of stars according to their classification would be the equivalent of:

> **1 star**: considered a good restaurant; **2 stars**: represents an excellent quality restaurant; **3 stars**: consists of an exceptional restaurant, of which it has extreme perfection; **Bib Gourmand**: This award is given to restaurants that, despite not being good enough to be awarded Michelin stars, are recognized for serving great quality food at moderate prices.

Therefore, *Michelin* stars certainly represent extreme importance for gastronomy, distinguishing restaurants based on very strict classifications. In this sense, the identification of restaurants with this classification is extremely important for them, in terms of their business model, and for lovers of good gastronomy.

In order to provide both a tool, in the form of an ontology and a general location distribution of restaurants with *Michelin* stars analysis, we will describe in this report how we created a restaurants ontology containing information from Michelin Guide with application in **France**, classifying restaurants with stars system. For that, we created an analysis from that ontology in a Jupyter notebook using Pandas Library for data engineering and visualization. This report will elaborate on the Goal and the Stakeholders of our project and provide an explanation and a

walkthrough of the designed analysis. Furthermore, it will describe our created ontology in detail, covering the Domain, Conceptualization, Inferences, Data Reuse and SPARQL queries.

2) Description of the Application and Users

Goal of the Analysis

The goal of our ontology is to deeply study *Michelin* Star awarded restaurants to enhance a client's decision on which restaurant to go. For the most part the clients that will be interested in our data will be international tourists.

Taking several factors into account, our unique ontology will allow someone to browse through *Michelin* Star restaurants located in France that match their desired cuisine type, budget, location, and other amenities. For example, if a client wanted to know what Michelin Star restaurants to visit in the city of Paris that are cheaper and have amenities like air-conditioning and a terrace, they would be able to do so.

Moreover, if someone wanted to know which *Michelin* Star restaurants in all of France are the most expensive and include the most amenities, they would be able to do so as well. While a future possibility of the study is to have an ontology that will be reusable for Michelin Star restaurants all around the world or for a specific country, city, or region, for this project, our analysis will focus exclusively on Michelin Star restaurants located in France.

In summary, this ontology should serve the purpose of helping tourists to make choices based on predefined criteria and helping restaurant entrepreneurs themselves to identify business and location opportunities.

Stakeholders

There are many potential stakeholders that could be interested in the data that will be contained within our ontology.

First, we refer to the Michelin Star restaurants located in France. These businesses have the most to gain from the creation of our ontology because our ontology will drive more business to these establishments and as a result restaurant profits will increase as their customer bases increase.

Secondly, we refer to customers, since they will now have access to an efficient recommendation system that will help them make better educated decisions when deciding on a food venue.

Ultimately, the success of this project will also benefit the French gastronomic industry, as it will aim to improve the experience of potential tourists.

In general, this project will clearly benefit tourists in their choices; it will help the restaurant sector and its entrepreneurs in the management of their businesses. In addition, nonetheless, it will help the French economy, which is heavily dependent on tourism.

3) Design and Walkthrough

Design

The design of the analysis started by looking for datasets and ontologies from external sources that were relevant and could be reused in the design. Regarding the topic, ontologies were searched for that contained and mapped information regarding the distribution of restaurants that were awarded Michelin Stars in French cities, which would be beneficial for the creation of the ontology. Datasets containing information about Michelin Star restaurants located in France would add vital data for the analysis to be accomplished.

As mentioned above, the overall goal of our analysis is to find the most suitable Michelin star restaurants to recommend to a potential customer according to a variety of factors. The metric of suitability responsible for that will consider the restaurant price range, it's number of Michelin stars and cuisine type specialties.

We will go about these respective design steps:

1. The first step to perform our analysis is to retrieve all the necessary information about *Michelin* star restaurants in France that are needed for our metric of suitability. This would mean for each *Michelin* star restaurant we find their name, average price, number of *Michelin* stars, geographical location, phone number, website, accepted payment methods and other various information that might become of significant importance later.

2. Once we have the required data, we then need to define our metric of suitability. This means we will have to find calculation to perform on our restaurants based on that data to rank which restaurant fits the best the needs of the user.

3. Once we have these visualizations, it is important to present to the interested parties (clients, restaurants, or tourist studies offices) significant visualizations in a clear, perceptible, and unambiguous way.

Those visualization plans enumerate as follow:

- Plot the number *Michelin* starred restaurants grouped by French city
- Present the distribution of the *Michelin* award according to its type by the different French restaurants
- Top cuisine type dishes offered by the *Michelin* starred restaurants

This project's analysis will be mainly based on 1 CSV retrieved from Kaggle. Obtained from the link: https://www.kaggle.com/datasets/ngshiheng/michelin-guide-restaurants-2021. As it will be mentioned along this report, this CSV dataset was found to be useful as it contained all the relevant attributes for the analysis. Some of its variables were in the wrong format and required data processing.

Walkthrough

The analysis will focus, as shown in the Jupyter Notebook file, on over 1000 Michelin award-winning restaurants after all data processing. After reading the README file, the reader will understand the purpose of each file belonging to this project. The course taken in carrying out this project will also be clear.

To start off with, the domain, goals, and questions to respond about this topic were chosen and outlined. Secondly, the part of data collection, cleaning and processing took place, with the data retrieval being done in Kaggle and all the data processing task with Pandas Python Library.

Then, with the main dataset (external source in CSV format) prepared, all the mapping decisions and correct construction and formalization of the ontology in OWL were taken. As such, it was possible to obtain important inferences, detailed throughout the project.

The last major area of the project will concern the extraction of knowledge, insertion of data using the second external source (DBPedia) and the production of visualizations responsible for clearly summarizing the proposed objectives of improving this branch of activity for all stakeholders, all done based on complex SPARQL queries.

4) Domain and Conceptualization

Domain and Scope of the Ontology

The overall goal of our ontology is to provide meaningful insights and guidance to potential clients who are interested in traveling to France for them to compile a list of Michelin Star restaurants they could visit that best fit their budget, cuisine, and interests. For this reason, the domain of our ontology would be all the Michelin Star restaurants in the country of France. This domain could be expanded upon further in the future with the introduction of new data from *Michelin* star restaurants in other countries.

In order to give a customer accurate recommendation according to their wants and needs, we will need to get specific information about each restaurant. For instance, if a customer wanted to visit only the cheapest *Michelin* star restaurants in France, we would need the data on the average price point for each restaurant. For each restaurant, we have the average price, cuisine, number of *Michelin* stars, and facilities. Facilities include what type of accepted payment methods the restaurant accepts or other miscellaneous attributes (wheelchair accessible, vegetarian options, etc.) Depending on what the customer is looking for, these specifications should allow us to give them accurate recommendations.

Methodology

This project's ontology is being designed and constructed using a bottom-up approach. The first thing that we did was figure out what *Michelin* star restaurant data that was available to us and that would also fit our needs.

This project's analysis will be mainly based on 1 CSV retrieved from Kaggle (*from the link: https://www.kaggle.com/datasets/ngshiheng/michelin-guide-restaurants-2021, which has as Collaborators "ngshiheng" (owner) and as author "Jerry Ng" bio: https://jerrynsh.com/about/*) that contained all the necessary information we need in order to achieve our goal:

name of the restaurant; address; minimum price; max price; accepted payment method; cuisine; longitude; latitude and the restaurants' telephone number.

Secondly, we proceeded to the Data Process phase, where we first filtered the data to only have restaurants located in France by stripping the Address column.

Thirdly, we renamed the column originally titled Location to City. Next, we removed the column Currency as every instance of our ontology will have euros as their accepted currency due to France's membership in the euro currency union.

Moving on, we noticed there were two restaurants without a phone number on one of the datasets. To tackle this issue, we proceeded to fetch these restaurants' phone numbers via the web and put them in the dataset. Finally, we split the Facilities column which was cluttered with various random information pertaining to each restaurant into relevant columns that could be useful for our project (types of payment accepted, accessibility, etc.). Once we were positive all the data was set, we then found *dbo* and *Schema.org* ontologies that would fit our data.

Once we had found the necessary classes and properties, we wanted to use from *dbo* and *Schema.org* we then had to map our data and merge the two ontologies together. We also created three classes ourselves that were subclasses of sch:Restaurant that we could use to make meaningful inferences. Those classes were CheapRestaurant, MiddlePricedRestaurant, and ExpensiveRestaurant.

This would allow us to infer a restaurant's price type based on the price properties. We however encountered an issue with mapping our data with onto-refine. The data properties that we needed to make class restrictions for our inferences for the three classes mentioned above, were showing up as annotation properties. This meant that we couldn't make the class restriction and make those inferences. Thankfully however solved this issue and we were able to redefine those annotation properties as object/data type properties. With that solved we had our data successfully mapped into RDF format, the ontology properly built in *protégé*, and we were able to combine the two into one final ontology.

There is a class in the ontology named Chef, which would ideally be populated using DBPedia as an external data source. However, due to the lack of a significant number of "Chef" entities in the same source, it was only possible to insert 3 distinct chefs. Thus, this class will be sparsely populated, which will create some constraints in terms of inference in this dimension of analysis.

Conceptualization

Our ontology is focused on *Michelin* star restaurants located in France. Because of this all, our classes and subclasses are designed to identify information regarding these restaurants in such a way that it is easy to extract valuable information on all the restaurants included within our dataset using the data properties we have designed.

Our ontology consists exactly of 19 classes, 10 subclasses, 4 object properties, and 8 data type properties. The main classes of our ontology are the following: Organization, Place, Intangible, City, Region, Award, Cuisine, CreativeWork, and Person.

These are especially important because they are the most encompassing classes and can be used to identify larger amounts of information.

Our subclasses:

[*LocalBusiness,*
FoodEstablishment,
Restaurant,
ExpensiveRestaurant,
MiddlePriceRestaurant,
CheapRestaurant,
Enumeration,
PaymentMethod,
Website, and
Chef]

Can help us identify more specific and unique information and are extremely helpful when paired with our several properties:

[*dbo:award,*
dbp:city,
dbp:cuisine,
sh:acceptedPaymentMethod,
rmf:maxPrice,
rmf:minPrice,
geo:lat,
geo:long,
sh:amenityFeature,
sh:priceRange,
sh:telephone,
sh:issn].

Ontology in detail

Classes and Subclasses (19 total) and Properties (12 total) – Ontology Schema

> Organization (1)
> LocalBusiness (2)
> FoodEstablishment (3)
> Restaurant (4)
> ExpensiveRestaurant (5)
> MiddlePriceRestaurant (6)
> CheapRestaurant (7)
> Place (8)
> Local Business
> Food Establishment
> Restaurant
> Intangible (9)
> Enumeration (10)
> Payment Method (11)
> City (12)
> Region (13)
> Award (14)
> Cuisine (15)
> CreativeWork (16)
> Website (17)
> Person (18)
> Chef (19)
Properties (12 total)
Object Properties:
> dbo:award (1)
> dbp:city (2)
> dbp:cuisine (3)
> sh:acceptedPaymentMethod (4)
Data Properties:
> rmf:maxPrice (5)
> rmf:minPrice (6)
> geo:lat (7)
> geo:long (8)
> sh:amenityFeature (9)
> sh:priceRange (10)
> sh:telephone (11)
> sh:issn (12)

Properties in detail

To maintain the consistency and fidelity of the ontology formalized in OWL, we proceeded to the definition of characteristics of both object and data properties, as well as the correct definition of their domains and ranges.

In more detail:

- The object property *dbp:award*, responsible for stating that a certain restaurant has won a certain *michelin* award, will be Asymmetric and Irreflexive and its domain and range will be *dbo:Restaurant* and *dbo:Award* respectively.

- The object property *dbp:city*, responsible for stating that a certain restaurant is located in a certain region, will be Asymmetric and Irreflexive and its domain and range will be *dbo:Restaurant* and *dbo:City* respectively.

- The object property *dbp:cuisine*, responsible for stating that a certain restaurant is specialized in a certain cuisine type, will be Asymmetric and Irreflexive and its domain and range will be *dbo:Restaurant* and *dbo:cuisine* respectively.

- The object property *sch:acceptedPaymentMethod*, responsible for stating that a certain restaurant accepts a certain payment method, will be Asymmetric and Irreflexive and its domain and range will be *dbo:Restaurant* and *sch:PaymentMethod* respectively.

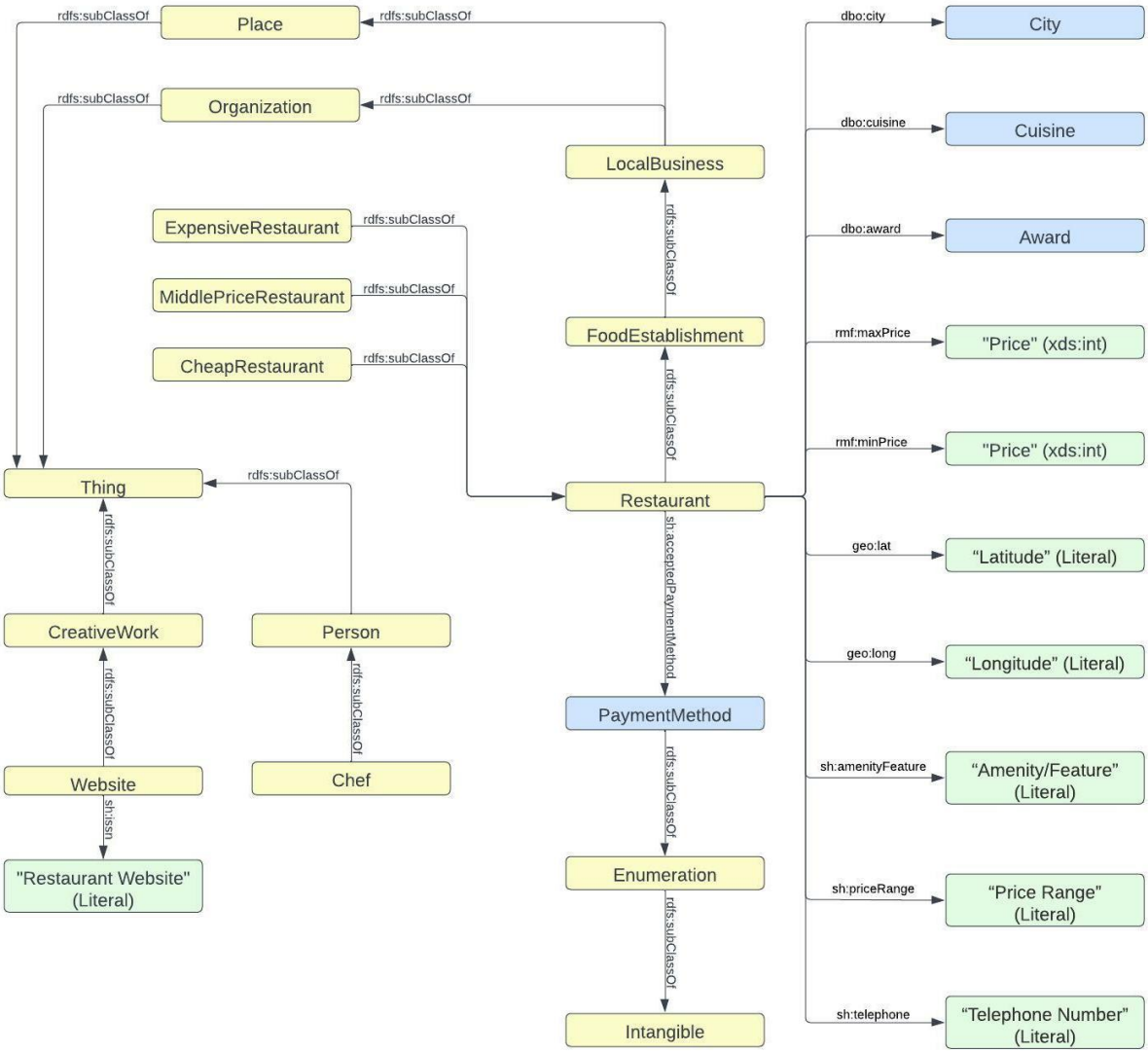
- The data properties *ex:maxPrice* and *ex:minPrice*, responsible for stating the minimum and maximum price charged in their menus, will have as domain *dbo:Restaurant* and as range *xsd:int.*

- The other data properties (*geo:lat, geo:long, sch:amenityFeature, sch:issn, sch:priceRange and sch:telephone)*, responsible for pointing at the restaurant's geographical coordinates, the amenitites it possesses, the restaurant's URL, their price range tag and their telephone number, will all have as domain instances of the class *dbo:Restaurant* and as range *rdf:PlainLiteral.*

The table below more clearly shows the relationships between the subjects and objects within our ontology when they are connected via our object properties. It also clearly shows the relationships between the subjects and the literal objects within our ontology when they are connected via our data type properties.

Subject	**Object Property**	**Object**
Restaurant	dbo:award	Award
Restaurant	dbo:city	City
Restaurant	dbo:cuisine	Cuisine
Restaurant	sh:acceptedPaymentMethod	PaymentMethod
Subject	**Data Type Property**	**Data Type**
Restaurant	rmf:maxPrice	xsd:int
Restaurant	rmf:minPrice	xsd:int
Restaurant	geo:lat	rdf:PlainLiteral
Restaurant	geo:long	rdf:PlainLiteral
Restaurant	sh:amenityFeature	rdf:PlainLiteral
Restaurant	sh:priceRange	rdf:PlainLiteral
Restaurant	sh:telephone	rdf:PlainLiteral
Website	sh:issn	rdf:PlainLiteral

Conceptualization Diagram of the Ontology



5) Ontology and Inferencing

Previously Existing Ontologies

Since we modeled and picked two previously existing ontologies (DBpedia & schema.org) that we will be using for our project around our data (bottom-up approach) we had a solid idea of how to map our data to our protégé ontology.

As our external dataset was formatted as a CSV file that we retrieved from Kaggle, we had to use onto-refine to establish the RDF mappings. However, before we could begin the process of RDF mapping, we had to split up some of the columns on our CSV file to make our data more useful. For example, we had a column titled "Facilities" which had all the important services/amenities the restaurant included (Good View, Terrace, Air-Conditioning, Wheelchair Accessible, etc..).

Moreover, since the CSV file contained restaurants from across the globe, we had to filter the French restaurants which was done using pandas and the column address from the CSV, where the last word of the address was the country. The country was extracted by splitting the address at commas and checking if the last column was "France".

We used python to split these "Facilities" into separate columns which would allow us to query that information accurately. Now that our CSV file was formatted correctly, we began to map out our ontology using onto-refine. Having said that, the defining principal entity for our ontology was a restaurant, we mapped each restaurant as belonging to the restaurant class on our ontology. We then assigned each restaurant the necessary properties like latitude, longitude, website URL, price range, necessary facilities and has been awarded with a certain *Michelin award.* We then continued to define the mapping between other classes like city, cuisine, website, and their corresponding properties.

Class Restrictions

For our ontology, we implemented the following 4 class restrictions:

- rmf:CheapRestaurant **owl:equivalentTo** ex:maxPrice some xsd:integer[<=100].
- rmf:MiddlePriceRestaurant **owl:equivalentTo** (ex:maxPrice some xsd:integer[<=200] and ex:minPrice some xsd:integer[>100]).
- rmf:ExpensiveRestaurant **owl:equivalentTo** ex:minPrice some xsd:integer[>200].
- dbo:Restaurant **owl:equivalentTo** sch:Restaurant.

For the process of creating the previously provided class restrictions we ended up reusing the existing **dbo** (DBPedia Ontology) and **sch** (schema.org) ontologies.

For the first restrictions set of class restrictions, our goal was to create new 3 price-categorized classes that will be populated in the future by restaurant instances dependently on their price charged.

A cheap *Michelin* star restaurant was considered cheap if its maximum price was lower or equal to 100€, while a middle-priced *Michelin* star restaurant would fall into this category if its minimum and maximum prices ranged from 100€ to 200€.

Ultimately, a *Michelin* star restaurant will be considered expensive if its minimum price is greater than 200€. These class restrictions were of utmost importance because it allowed us to make meaningful inferences of the most accurate recommendations based on the user's budget which is known to be a key factor on the decision taking.

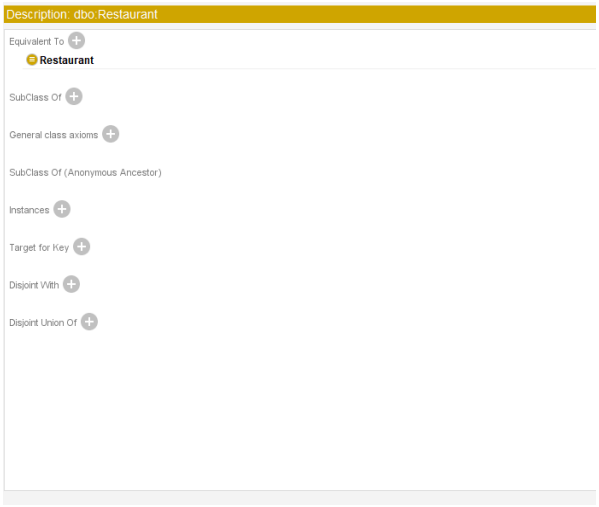
Meaningful Inferences

Our ontology was able to create some meaningful and useful inferences. One of the most important inferences was ensuring that our sch:Restaurant class which we mapped all our restaurants to was owl:EquivalentClass to dbo:Restaurant. This would mean that all restaurants that were mapped as *schema.org* were also valid for *dbo* which would allow us to query on the SPARQL DBPedia endpoint for more restaurant's insights. Below is a photo of dbo:Restaurant without the reasoner turned on, and then next to it is all the classes inferred with the reasoner on.

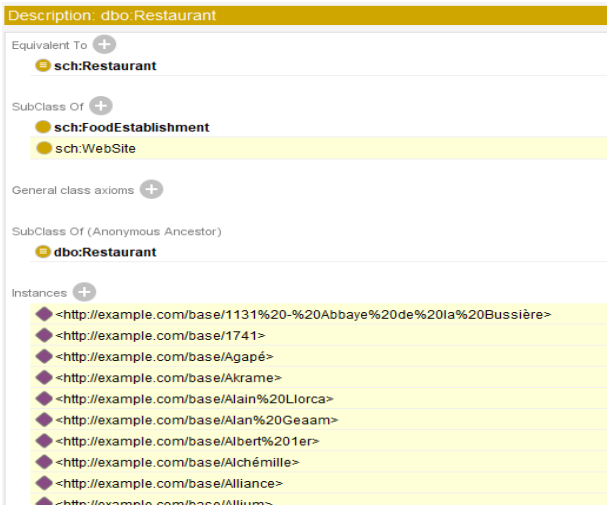
After a careful look, it is possible to conclude that the dbo:Restaurant class was inferred by the reasoner to be a subclass of the sch:WebSite class. It may seem an unwanted inference, however, it is not. One of the purposes of this analysis is to serve as a client's recommendation system and, therefore, for the purpose of presenting a particular restaurant recommendation, it makes sense to assign it the subclass relationship to the sch:WebSite class.

This means that each instance of type dbo:Restaurant class will be an instance of type sch:WebSite class, facilitating customer access to the restaurant's URI, enabling reservations, complaints or phone calls.

Restaurants Websites Inferences



Reasoner Off

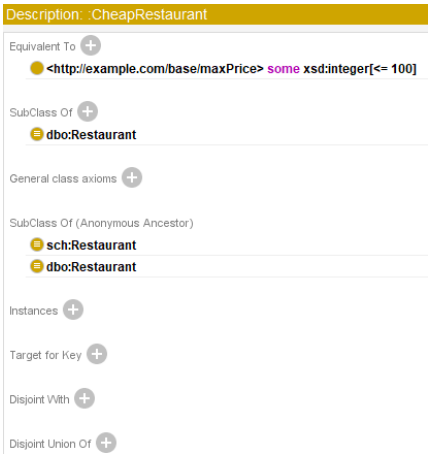


Reasoner On

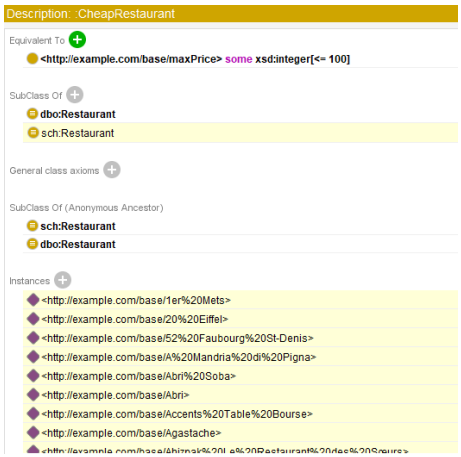
The next inferences that were made to the restaurant was on which price category they would be placed. In order to provide more useful ways to query the restaurants, we created three classes that were all subclasses of sch:Restaurant. These were: rmf:CheapRestaurant, rmf:MiddlePriceRestaurant, and rmf:ExpensiveRestaurant.

This would allow us to make inferences about a restaurant depending on its price point which would ultimately be helpful for a situation such as when a customer wants to decide how much they are comfortable to spend at a restaurant. These inferences are based on class restrictions. If you wish to view the exact specifications for what determines a cheap, middle priced, or expensive restaurant, see the class restrictions section on the previous page. Pictured below are the inferences for the class cheap restaurant. Following that are the inferences for the middle-priced restaurant. Lastly the inferences for the expensive restaurant are pictured.

Cheap Restaurants Inferences

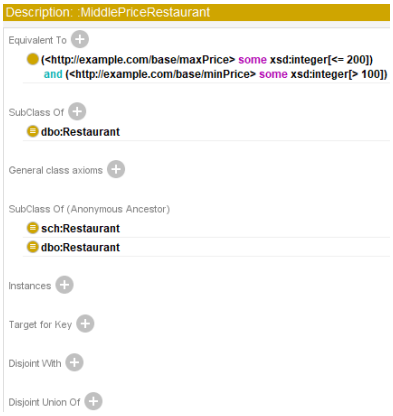


Reasoner Off

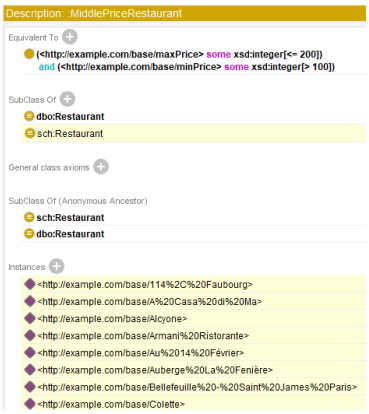


Reasoner On

Middle-Priced Restaurant Inference

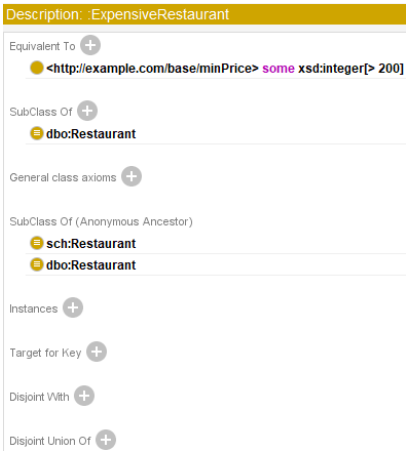


Reasoner Off

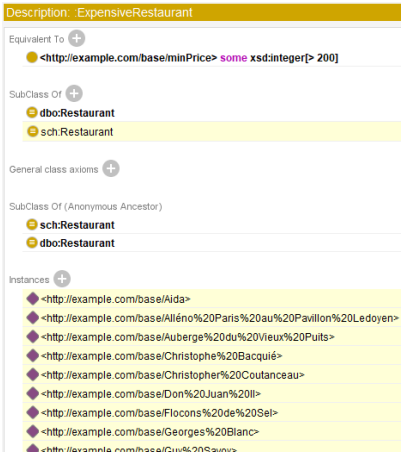


Reasoner On

Expensive Restaurants Inference



Reasoner Off



Reasoner On

6) Data Reuse

Regarding the data reuse phase, it is first important to identify the two datasets obtained externally. The first concerns the csv file (https://www.kaggle.com/datasets/ngshiheng/michelin-guide-restaurants-2021) which contained crucial information about each restaurant such as its name, geographic locations, minimum price charged, maximum price charged, website, telephone number and type of specialized cuisine and *Michelin* awards obtained, meaningful for the development of our project.

This dataset originally had 14 columns with the information referred above and 6527 rows each referring to a particular restaurant. Subsequently, as mentioned in the Data Engineering section, this dataset was subjected to processing operations to extract only the data necessary for this project, specifically, only the restaurants located in French territory. Operations done, this dataset was ready to be used with 13 columns and 1066 rows.

	Name	City	MinPrice	MaxPrice	Cuisine	Longitude	Latitude	PhoneNumber	Url	WebsiteUrl	Award
0	Le_Pré_Catelan	Paris	150	330	Creative, Modern Cuisine	2.250718	48.863937	3.314414e+10	https://guide.michelin.com/en/ile-de-france/pa...	https://www.leprecatelan.com/	3 MICHELIN Stars
1	Alléno_Paris_au_Pavillon_Ledoyen	Paris	280	395	Creative	2.316499	48.866093	3.315305e+10	https://guide.michelin.com/en/ile-de-france/pa...	https://www.yannick-alleno.com	3 MICHELIN Stars
2	Arpège	Paris	185	460	Creative	2.316965	48.855731	3.314705e+10	https://guide.michelin.com/en/ile-de-france/pa...	https://www.alain-passard.com/	3 MICHELIN Stars
3	Épicure	Paris	350	395	Modern Cuisine	2.314366	48.871729	3.315343e+10	https://guide.michelin.com/en/ile-de-france/pa...	https://www.oetkercollection.com/fr/hotels/le-...	3 MICHELIN Stars
4	Plénitude_-_Cheval_Blanc_Paris	Paris	330	405	Creative	2.342159	48.858815	3.317936e+10	https://guide.michelin.com/en/ile-de-france/pa...	https://www.chevalblanc.com/en/	3 MICHELIN Stars
...	...	...	...	...	...	...	...	...	...	...	...
1061	La_Ferme_de_la_Haute_Crémonville	Saint-Étienne-du-Vauvray	33	55	Traditional Cuisine	1.200692	49.240624	3.323259e+10	https://guide.michelin.com/en/normandie/saint-...	https://www.lafermedelahautecremonville.com/	Bib Gourmand
1062	Auberge_le_Nézil	Saint-Lyphard	35	43	Modern Cuisine	-2.327502	47.371565	3.324091e+10	https://guide.michelin.com/en/pays-de-la-loire...	https://aubergelenezil.fr/	Bib Gourmand
1063	Le_6_à_Table	Caromb	56	56	Modern Cuisine	5.107207	44.110128	3.349062e+10	https://guide.michelin.com/en/provence-alpes-c...	https://www.pascal-poulain.com/fr/	Bib Gourmand
1064	Arkadia	Vallon-Pont-d'Arc	21	47	Creative	4.394904	44.407909	3.362077e+10	https://guide.michelin.com/en/auvergne-rhone-a...	https://arkadia-restaurant.business.site/	Bib Gourmand

Pandas Dataset structure

The second external data source will be DBPedia, accessed using a SPARQL Endpoint. This decision is related to the need to populate the Chef class in the ontology formalized in OWL. We will execute the automatic population process of the Chef class using federated queries linked to our OWL ontology and using the following endpoint: "<https://dbpedia.org/sparql>".

Finally, we will take advantage of the different properties and classes that DBPedia has to offer, such as: ?chef rdf:type, dbo:Chef, dbp:chef, dbp:headChef, dbo:headChef, dbo:chef.

Description about the DBPedia's Chef entity

The insertion will be possible due to the presence of the properties mentioned above in the restaurants DBPedia page. Here follows an example of a restaurant present in CSV external data source and also present in the DBPedia external data source and its relevant properties:

dbp:chef	• Franck Pelux (en)
dbp:city	• dbr:Strasbourg
dbp:country	• dbr:France
dbp:currentOwner	• Cedric Moulot (en)
dbp:established	• 1971 (xsd:integer)
dbp:foodType	• dbr:French_cuisine

The query and the successful output responsible for that insertion process is as follows:

```
PREFIX dbp: <http://dbpedia.org/property/>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX sch: <https://schema.org/>

insert {?chef rdf:type dbo:Chef}  where {
?rmfRestaurant rdf:type sch:Restaurant ;
     rdfs:label ?rmfRestaurantLabel .
SERVICE <https://dbpedia.org/sparql> {
    {?dbpRestaurant rdf:type dbo:Restaurant .}
    {?dbpRestaurant rdfs:label ?dbpRestaurant_label .}
    {?dbpRestaurant dbp:chef ?chef .}
    union
    { ?dbpRestaurant dbp:headChef ?chef .}
    union
    { ?dbpRestaurant dbo:headChef ?chef .}
    union
    { ?dbpRestaurant dbo:chef ?chef .}
}
FILTER(?rmfRestaurantLabel = str(?dbpRestaurant_label))
}
```

Added 3 statements. Update took 2m 3s, minutes ago.

Here follows the different URI's of the founded chefs:

Filter query results

Showing results from 1 to 3 of 3. Query took 0.1s, today at 18:15.

	chef
1	http://dbpedia.org/resource/Mauro_Colagreco
2	http://dbpedia.org/resource/Frédéric_Anton
3	http://dbpedia.org/resource/Jos_P._Boomgaardt

7) Queries

SPARQL Queries

In our project, SPARQL Queries are used to display the data about the *Michelin* Star awarded restaurants in France, based on the required analysis. This analysis will then be displayed through Pandas plots in an easy and perceptible way. All SPARQL queries that bring valuable information to the analysis are shown below.

We will proceed to the presentation and explanation of 6 queries performed.

1) Cities grouped by total restaurants, from the city that has the most restaurants to the least.

Code:

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbr: <http://dbpedia.org/resource/>
PREFIX dbp: <http://dbpedia.org/property/>
PREFIX sh: <https://schema.org/>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>

select ?city_name (count(?r) as ?total_restaurants) where {

    ?r dbp:city ?c .

    ?c rdf:type dbo:City;

       rdfs:label ?city_name.

}
group by ?city_name

order by desc(?total_restaurants)
```

Outcome:

Filter query results — Showing results from 1 to 638 of 638. Query took 0.8s, minutes ago.

	city_name	total_restaurants
1	"Paris"	"165"^^xsd:integer
2	"Lyon"	"27"^^xsd:integer
3	"Clermont-Ferrand"	"11"^^xsd:integer
4	"Nice"	"11"^^xsd:integer
5	"Bordeaux"	"10"^^xsd:integer
6	"Courchevel"	"9"^^xsd:integer
7	"Dijon"	"9"^^xsd:integer
8	"Toulouse"	"9"^^xsd:integer

2) For each *Michelin* award attribution in the dataset, the number of restaurants awarded with each award, sorted in descending order of the total restaurants. This helps in understanding the most dominant and least dominant award type and the number of restaurants they account for.

Code:

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbp: <http://dbpedia.org/property/>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

select ?award (count(?s) as ?total_restaurants) where {

    ?s rdf:type    dbo:Restaurant;
    dbp:award ?a.
     ?a rdfs:label ?award.
}
group by ?award
order by desc(?total_restaurants)
```

Outcome:

Filter query results — Showing results from 1 to 4 of 4. Query took 0.1s, minutes ago.

	award	total_restaurants
1	"1 MICHELIN Star"	"511"^^xsd:integer
2	"Bib Gourmand"	"437"^^xsd:integer
3	"2 MICHELIN Stars"	"74"^^xsd:integer
4	"3 MICHELIN Stars"	"31"^^xsd:integer

3) Cuisine types grouped by number of restaurants specialized in a certain cuisine, from the most popular cuisine to the least popular one. It helps in understanding the popularity of a cuisine in the French Michelin restaurants.

Code:

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbr: <http://dbpedia.org/resource/>
PREFIX dbp: <http://dbpedia.org/property/>
PREFIX sh: <https://schema.org/>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>

select ?cuisine_type (count(?res) as ?total_restaurants) where {

    ?res dbp:cuisine ?cui.

    ?cui rdfs:label ?cuisine_type.

}

group by ?cuisine_type

order by desc(?total_restaurants)
```

Outcome:

Filter query results — Showing results from 1 to 83 of 83. Query took 0.2s, minutes ago.

	cuisine_type	total_restaurants
1	"Modern Cuisine"	"545"^^xsd:integer
2	"Creative"	"139"^^xsd:integer
3	"Traditional Cuisine"	"89"^^xsd:integer
4	"Classic Cuisine"	"35"^^xsd:integer
5	"Modern Cuisine, Creative"	"29"^^xsd:integer
6	"Creative, Modern Cuisine"	"24"^^xsd:integer
7	"Market Cuisine"	"18"^^xsd:integer
8	"Seafood"	"13"^^xsd:integer

4) Minimum "maximum price" practiced in € by a restaurant in a single item and its identification.

Code:

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ex: <http://example.com/base/>

select ?restaurant_label ?price where{

?res rdf:type dbo:Restaurant.

?res ex:maxPrice ?price.

?res rdfs:label ?restaurant_label

{select (min(?p) as ?max_price) where{

?res ex:maxPrice ?p

    }}
filter(?price=?max_price)
}
```

Outcome:

Filter query results — Showing results from 1 to 1 of 1. Query took 0.4s, moments ago.

	restaurant_label	price
1	"La Petite Ourse"	"25"^^xsd:int

5) Absolute maximum price in € in the database and the restaurant's identification

Code:

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ex: <http://example.com/base/>

select ?restaurant_label ?price where{

    ?res rdf:type dbo:Restaurant.

    ?res ex:maxPrice ?price.

    ?res rdfs:label ?restaurant_label

    {select (max(?p) as ?max_price) where{

    ?res ex:maxPrice ?p

        }}

    filter(?price=?max_price)
}
```

Outcome:

Filter query results — Showing results from 1 to 1 of 1. Query took 0.2s, minutes ago.

	restaurant_label	price
1	"L'Écrin"	"735"^^xsd:int

6) A list of all the restaurants and their location with the minimum price and maximum price ranging from 200€ to 550€, ordered ascending by minimum price and as there are many cases of "draws" in the minimum prices, these cases were ordered by maximum price in a descending order so that the most expensive would come first. If a client has a budget, this query extracts all the affordable restaurants for a certain budget given a minimum and maximum price. In the query, the minimum price was set to 200€ and the maximum price to 550€.

Code:

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbr: <http://dbpedia.org/resource/>
PREFIX dbp: <http://dbpedia.org/property/>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ex: <http://example.com/base/>

select ?restaurant_label ?city_label ?min_price ?max_price where{
        ?res rdf:type dbo:Restaurant;
         rdfs:label ?restaurant_label;
         dbp:city ?city.
    ?city rdfs:label ?city_label.
    ?res ex:minPrice ?min_price.
    ?res ex:maxPrice ?max_price
    filter(?min_price>200)
    filter(?max_price<550)

}

order by asc(?min_price) desc(?max_price)
```

Outcome:

Filter query results — Showing results from 1 to 28 of 28. Query took 0.2s, minutes ago.

	restaurant_label	city_label	min_price	max_price
1	"Auberge du Vieux Puits"	"Fontjoncouse"	"205"^^xsd:int	"235"^^xsd:int
2	"Yoann Conte"	"Veyrier-du-Lac"	"210"^^xsd:int	"350"^^xsd:int
3	"Christopher Coutanceau"	"La Rochelle"	"210"^^xsd:int	"280"^^xsd:int
4	"Maison Nouvelle"	"Bordeaux"	"210"^^xsd:int	"210"^^xsd:int
5	"Don Juan II"	"Paris"	"220"^^xsd:int	"220"^^xsd:int
6	"Flocons de Sel"	"Megève"	"230"^^xsd:int	"310"^^xsd:int
7	"Le Montgomerie"	"Courchevel"	"230"^^xsd:int	"310"^^xsd:int
8	"Jean Sulpice"	"Talloires-Montmin"	"235"^^xsd:int	"275"^^xsd:int

SPARQL querying over the ontology's inferences

Knowing now that our dataset can successfully respond to SPARQL queries, it will be possible to make valuable inferences about the ontology.

The first will be to obtain the averages of the maximum prices practiced in each of the inferred classes (*CheapRestaurants*, *MiddlePriceRestaurants*, *ExpensiveRestaurants*). It will be an inference of the utmost importance as it will allow our recommendation system to narrow down a given customer's restaurant recommendation list according to their budget, increasing the chances of customer satisfaction.

Code:

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ex: <http://example.com/base/>
PREFIX rmf: <http://www.co-ode.org/ontologies/ont.owl#>

select (avg(?p) as ?average_min_price_cheap_restaurants)
    where{
    ?restaurant a dbo:Restaurant;
                a rmf:CheapRestaurant;
                rdfs:label ?res_label;
                ex:maxPrice ?p

}
```

Outcome:

	average_max_price_cheap_restaurants
1	"66.032815198618307426597582"^^xsd:decimal

Average Maximum Price for the Cheap Restaurants ~ 66.03€

Code:

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ex: <http://example.com/base/>
PREFIX rmf: <http://www.co-ode.org/ontologies/ont.owl#>

select (avg(?p) as ?average_max_price_middle_restaurants)
    where{
    ?restaurant a dbo:Restaurant;
                a rmf:MiddlePriceRestaurant;
                rdfs:label ?res_label;
                ex:maxPrice ?p

}
```

Outcome:

	average_max_price_middle_restaurants
1	"157.92063492063492063492063 5"^^xsd:decimal

Average Maximum Price for the Middle-Priced Restaurants ~ 157.92€

Code:

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ex: <http://example.com/base/>
PREFIX rmf: <http://www.co-ode.org/ontologies/ont.owl#>

select (avg(?p) as ?average_max_price_expensive_restaurants)
    where{
    ?restaurant a dbo:Restaurant;
                a rmf:ExpensiveRestaurant;
                rdfs:label ?res_label;
                ex:maxPrice ?p
}
```

Outcome:

	average_max_price_expensive_restaurants
1	"341.89655172413793103448275 9"^^xsd:decimal

Average Maximum Price for the Expensive Restaurants ~ 341.90€

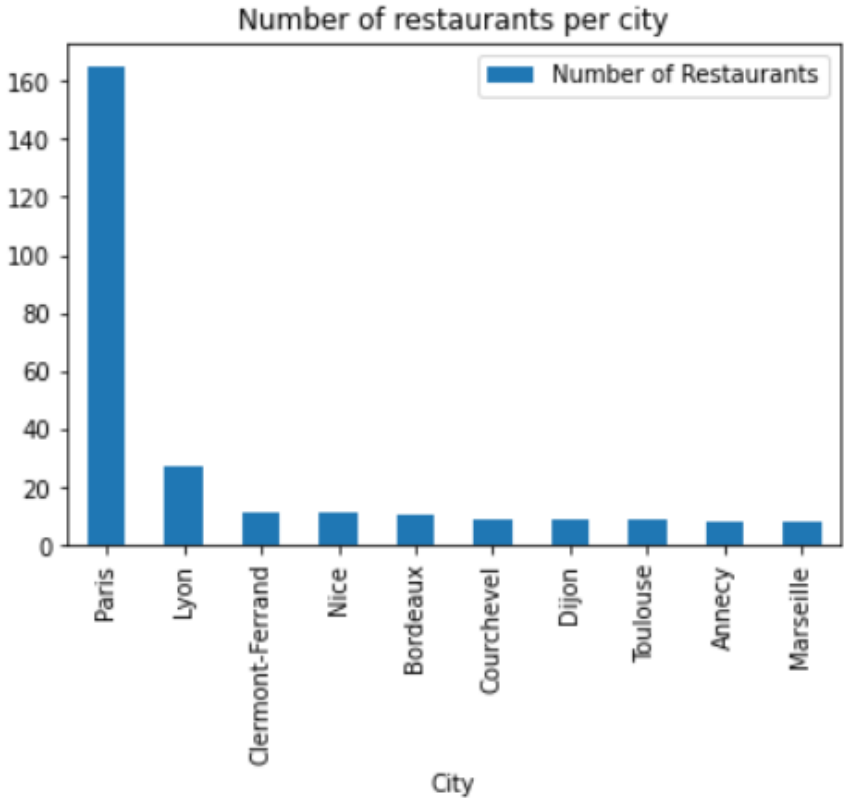
There is a clear trend, meaning that this categorization of restaurants according to their average of average prices charged will produce predictive gains.

8) Honorary title: Visualization Guru

For this project, we will pursue the honorary title of Visualization Guru. As such, we produced several interesting graphics given the domain and commitment of the project in question, to generate new knowledge with a view to benefiting stakeholders. All visualizations were performed in the notebook produced in Python language using the Pandas library.

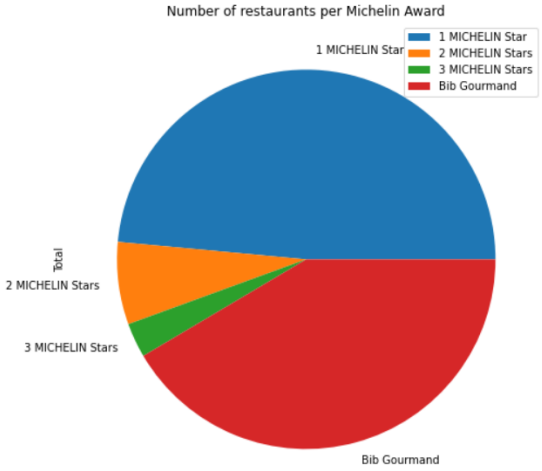
To start with, we present a bar graph relating cities to the total number of restaurants that hold any *Michelin* award. For the purposes of simplicity, only the 10 cities with the most restaurants are shown. We consider this information to be relevant for the French Government Ministry of Tourism.

It is possible to conclude that most of these restaurants are in Paris. A good strategy to reduce this centralism of demand for gastronomic tourism in the capital Paris would be to provide incentives for opening and investing in restaurants of this nature in other cities or regions of the country.

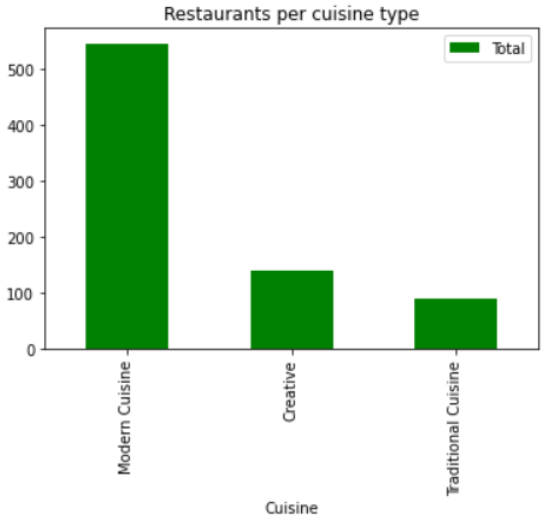


Secondly, to better understand the distribution of *Michelin* awards among the various French restaurants, a pie chart was created relating each Michelin award to the total number of restaurants awarded that award. It is important here to explain the meaning of the Bib Gourmand award. This award is given to restaurants that, despite not being good enough to be awarded *Michelin* stars, are recognized for serving great quality food at moderate prices.

Having observed the graph it is possible to conclude that attribution of prizes practically always adjusts to the "rarity" of each one of them. The most common award is 1 *Michelin* star, followed by Bib Gourmand (an unexpected fact), followed by 2 *Michelin* stars and finally, as the rarest award, 3 stars, as expected.



Thirdly, it was considered relevant to find out, for the purposes of popularity of demand and possible financial impacts on the most profitable types of business, which was the top 3 of the most popular types of cuisine in the target restaurants. It was concluded from the following chart that customers show a preference for the types of Modern, Creative and Traditional cuisines, with the Modern style cuisine being the most requested.

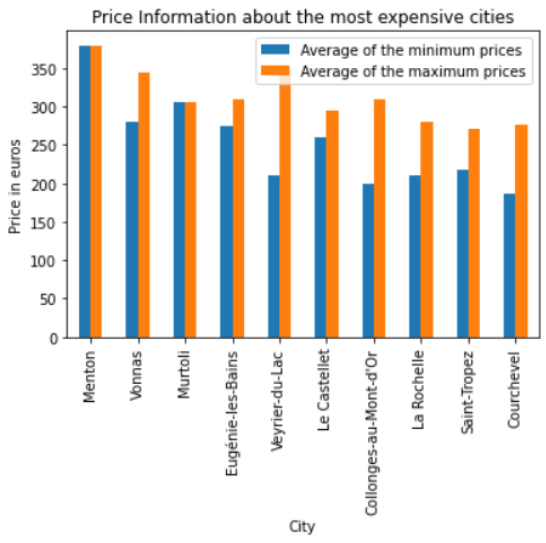


To finish off with, and now in a more complex way, we moved on to the correlation crossing of two ontology variables and their visualization. These are the cities and the prices charged in their restaurants. To begin with, we established the following as a mathematical criterion for comparing cities in the most expensive/cheapest ratio:

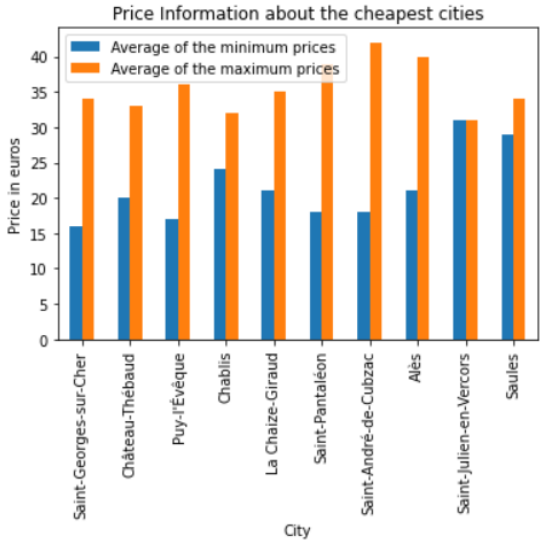
(average of the minimum price charged + average of the maximum price charged) / 2

Subsequently, we proceeded to the construction of two bar graphs, comparing the average of the maximum price practiced with the average of the minimum price practiced. The first of them was applied to the 10 most expensive cities and the second to the 10 cheapest cities.

This algorithm will prove to be important because by allowing a comparison between cities at the level of average prices, it will be useful to, when questioning the user of their maximum or range of money willing to spend and the city where they are in, present the best recommendation of restaurants possible.



Prince insights about the 10 most expensive cities



Price insights about the 10 cheapest cities