

# **UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA**

Dipartimento di Scienze Fisiche, Informatiche e  
Matematiche

---

Corso di Laurea in Informatica

## **Exploratory Data Analysis & Data Engineering Techniques for the study of Copy Number Variations**

**Relatore:**

Chiar.ma Prof.ssa Federica Mandreoli

**Candidato:**

Fabio Bove

**Correlatori:**

Chiar.mo Prof. Cristian Taccioli

Chiar.ma Dott.ssa. Chiara Vischioni

Chiar.mo Dott. Riccardo Martoglia

---

**Anno Accademico 2019/2020**



# Index

<b>1 Introduction .....</b>	<b>4</b>
1.1 A first look at the study.....	5
1.1.1 Copy Number Variations, Parameter of interest.....	7
1.1.2 Scientific Utility.....	9
<b>2 A platform for researchers.....</b>	<b>11</b>
2.1 Dataset Properties .....	12
2.1.1 Organisms, Gene and Genes Family characteristics.....	14
2.2 Platform Schema.....	15
<b>3 Planning Choices .....</b>	<b>16</b>
<b>4 Descriptive Analysis Models .....</b>	<b>18</b>
4.1 Common Genes Model .....	22
4.2 Gene Quantity Model.....	27
4.3 Genes Intersection Model .....	29
<b>5 Exploratory Data Analysis .....</b>	<b>32</b>
5.1 Operation of the Exploratory Data Analysis.....	34
<b>6 Data Visualization .....</b>	<b>37</b>
<b>7 Conclusions .....</b>	<b>40</b>

## **1 Introduction**

The study is born from the collaboration between the University of Modena and the University of Padova, with the proposal of Prof. Cristian Taccioli (Professor of Molecular Biology and Applied Bioinformatics) and Chiara Vischioni (PhD student at the Department of Animal Medicine, Production and Health) to examine the genome of multiple species in order to obtain useful information for the research, dealing with tumor suppressor and oncogenic genes.

The aim of this project is to create a database that combines Copy Number Variations (CNVs) data among different organisms with other information related to their cancer rates, longevity and vital parameters.

The information will be retrieved from public on-line genomic libraries, that will allow, for the first time, to realize a complete collection of the CNVs within multiple species

In addition, for a small number of organisms with known cancer mortality rate and longevity, descriptive analysis models, coped with statistical measurements, will be carried out in order to enable researchers to identify particular trends or patterns, highlighting which are the genes linked to cancer resistant species or long-living once.

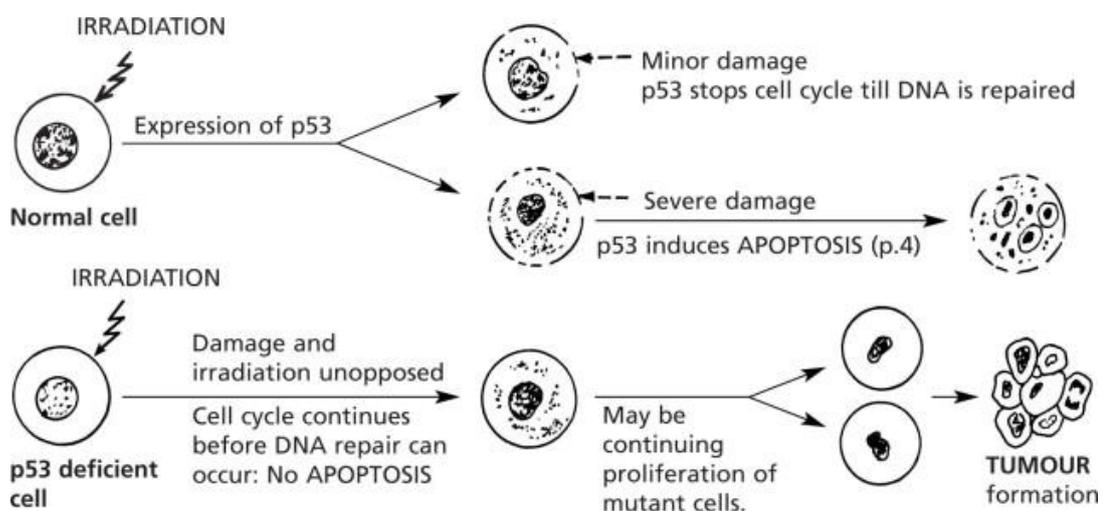
The application will also show possible correlations between cancer and other data, such as species metabolism, average weight of an organism or genes' families abundance in their genome.

## 1.1 A first look at the study

Recently, cancer research has looked at the study of those organisms with particular properties in terms of cancer resistance and high longevity rates, producing fundamental discoveries about which are the tumor resistance mechanisms and genes that most protect disease development. The study of a phenomenon as complex as cancer, that depends on several factors still unexplored, needs to combine different data both on organisms characteristics and genes function, in order to better observe which are the trends that could have an interesting behavior.

More advanced studies are taking advantage of those organisms with known low cancer rate, in order to detect the defensive mechanisms put in place and to identify through the analysis of their genetic heritage, genes in multiple copies or mutations, that can slow down or completely eliminate the onset of cancer.

Tumor suppressors (TS) are, genes that can slow down the cellular division mechanism; they are also able to repair DNA replications errors and start apoptosis, through with cell death is programmed. A malfunction of a tumor suppressor, due to its mutation for example, can lead to the proliferation of cells without any control causing the development of cancer *Fig.1.1*.



[Fig.1.1] Protein p53 function in response to cellular damage, e.g. irradiation. [1]

On the contrary, an oncogene usually promotes cell growth. Over-expression, mutations, or multi-copy presence of it, can, sometimes, lead to the onset of cancer.

Studying organisms which are prone to cancer is therefore useful, in order to find tumor suppressors and oncogenes mutations: down-regulation and over-expression. For example, we can investigate TS gene that is inactivated or that presents a reduced number of copies compared to the average, assuming in this case, a higher probability of cancer development.

To quote an article on mammalian cancer resistance mechanisms, I would like to emphasize the need to broaden the spectrum of research and to make clear how useful is the tool that we have worked so hard on: *“Cancer researchers have traditionally used the mouse and the rat as staple model organisms. These animals are very short-lived, reproduce rapidly and are highly prone to cancer. They have been very useful for modelling some human cancer types and testing experimental treatments; however, these cancer-prone species offer little for understanding the mechanisms of cancer resistance.”* [2]

The ability to visualize and manage the interactions between different variables involved in cancer onset represents a further step forward in research. Our tool will provide researchers and biologist with the appropriate knowledge, an excellent samples models, such as new potential oncogenes and tumor suppressors.

In addition, as highlighted by numerous articles in the biological field: *“Understanding the molecular mechanisms of cancer resistance in all these species is important and timely, as, ultimately, these mechanisms could be harnessed for the development of human cancer therapies.”* [2]

### 1.1.1 Copy Number Variations, Parameter of interest

The principal variable considered in our study is the variation in the number of gene copies or gene families between the genome of different organisms [192]. This is a phenomenon whereby a gene may be present in multiple or reduced copies within an individual's genome.

According to H.Dear, *“Copy-number variants have already been associated with some diseases and disease susceptibilities and are likely to prove as significant as sequence polymorphisms in this respect. Changes in copy number of parts of the genome are known to be a feature of many cancers, and their analysis is expected to reveal genes involved in carcinogenesis.”* [3]

Therefore, multiple copies of a gene in one organism compared to another can be associated with the increased risk of diseases such as cancer or can lead to particular phenotypic manifestations of a character, furthermore, *“longevity-associated gene families have a marginally significantly higher rate of duplication compared to non-longevity-associated gene families.[...] The identification of a gene cluster that duplicated solely in long-lived species involved in such fundamental processes provides a promising avenue for further exploration of Eutherian longevity evolution.”* [4]

Cancer research is not the only area in which CNVs plays a role. Other researchers, dealing with longevity for example, faced diseases related to cell reproduction.

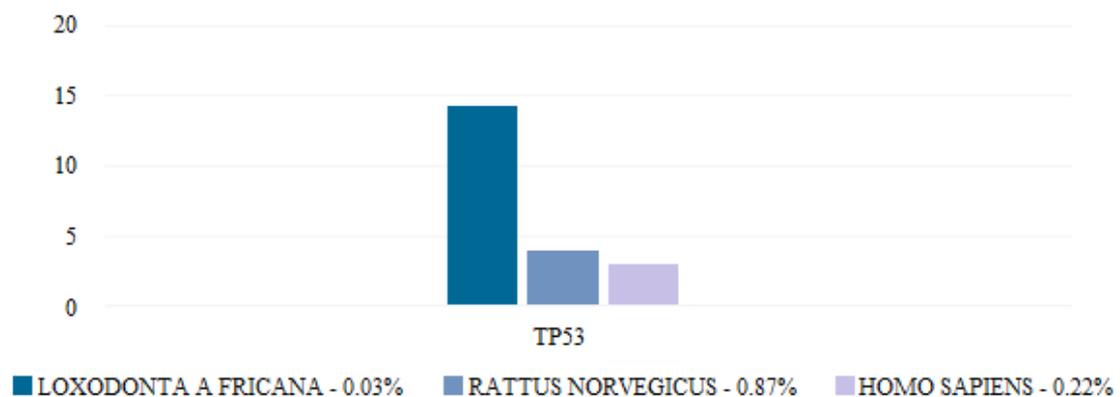
Reporting Magalh's considerations: *“Anti-longevity-associated gene families have significantly increased rate of duplication compared to pro-longevity gene families and are enriched in neurodegenerative disease categories. Conversely, duplicated pro-longevity-associated gene families are enriched in cell cycle genes”* [4]

Longevity and cancer can, therefore, be combined and simultaneously studied, allowing us to gain a wider view of healthy aging process.

An example of the analysis of cancer in long-lived mammals is the research carried out by Schiffman's team and his collaborator, with the aim to uncover the elephants' secrets to cancer resistance, as reported from *Vivian Callier* in an article about Peto's Paradox - [5].

The elephant, indeed, takes part of a very particular category of organisms. One would expect that *"the risk of developing cancer should theoretically increase with both the number of cells and the life span of an organism. However, gigantic animals do not get more cancer than humans, suggesting that super-human cancer suppression has evolved numerous times across the tree of life. This is the essence and promise of Peto's Paradox."* [6]

The researchers scoured the elephant genome, discovering extra copies of the tumor suppressor gene TP53, which is also present in humans and most other animals.



[Fig.1.2] TP53 Copies in Elephant, Mouse and Human, alongside with their cancer death percentage.

This gene detects irreparable DNA damage that could make a cell cancerous and once found the p53 protein triggers cell death, for example people born with a mutation in TP53 develop Li-Fraumeni syndrome and have a lifetime risk of developing cancer approaching 100%.

### 1.1.2 Scientific Utility

The identification of the molecular mechanisms responsible for the occurrence of tumors identifies the targets to which attention should be directed, identifying therapies targeted to the anomalies that specifically distinguish neoplastic cells from healthy ones.

In some experiments it has been shown how the specific inhibition of the enzyme of an oncogene is able to completely block the pathological activity of its protein and induce the regression of the tumors in which it is involved, *“one of the most encouraging examples is represented by the inhibitors of the Ras oncogene, which is activated in 20 ÷ 30% of tumors.”* [7]

Numerous genes are present in reduce copies or are down regulated in cancer cells: those are tumor suppressors and genes that code enzymes for DNA repair. These are destined to arouse increasing interest, because they are responsible for the predisposition to the onset of cancer as heritable.

An ideal prevention activity should aim at the recognition of the causes of tumor prior to his development, for example, with the advancement of research *“comparative studies could highlight potential targets where the genetic mechanisms underlying cancer suppression in one species could be transferred to an-other, with clinical implications. For instance, it was found that genetically altering mice to overexpress a form of the TP53 protein conferred a cancer-suppressive phenotype; however, these mice also displayed a premature ageing phenotype. Surprisingly, another study created ‘super p53’ mice which contained extra copies of the TP53 gene—similar to the elephant genome—under their normal promoters, and these mice revealed an enhanced DNA damage response and cancer suppression without the ageing effect. Work is now underway to develop medicines based on the TP53 pathway.”* [6]

This approach is defined in the complex 'gene therapy' and is today the most ambitious goal of molecular medicine. It consists in artificially reconstructing the genetic information necessary for the synthesis of the protein and replacing it with the damaged gene, with a sort of molecular surgery.

The fact that the platform we have created has allowed researchers in Padova to identify some possible tumor suppressor genes, which will be further studied with appropriate laboratory tests, is for us a source of great satisfaction and represents the concrete possibility to realize drugs for the direct attack on this evil.

## **2 A platform for researchers**

The first phase of the development starts with a meeting with researchers in Padova where is immediately highlighted the need to collect an important amount of information, mainly concerning Genes and Organisms, able to grow over time in the face of new discoveries in the scientific-biological field.

We also need a system that allows researchers to perform out in-depth searches on the data in our possession in order to verify the accuracy of scientific hypotheses and to visualize different reports for groups of samples, Genes or Organisms, considered interesting for cancer research.

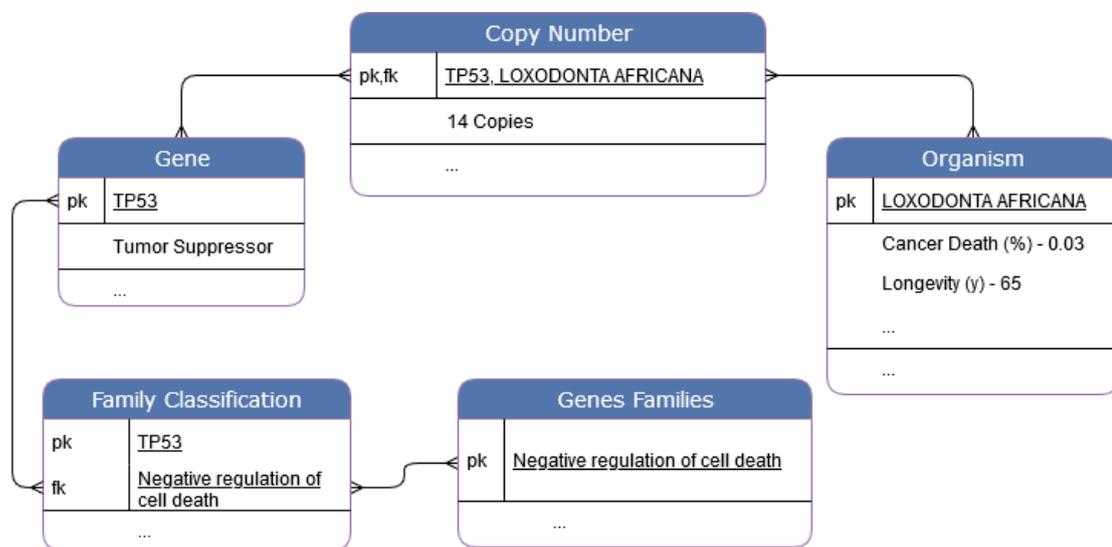
From the outset therefore we determined that the tools that could satisfy all the needs evidenced by the researchers was a Web Application, with the support of a Database to store various information and their relationships. In addition, different Descriptive Analysis Models will be implemented, by combining different queries they will provide relevant portions of data.

The next step involves the research and study of the different types of data visualization, as well as a graphic library, for the generation of complex graphs capable of showing large amounts of data purchases effectively. Data Visualization is important to highlight the relationships that exist between several elements and according to the data requested, all within a reasonable time.

## 2.1 Dataset Properties

The most important properties of our study are the combination, for the first time, of data concerning organisms and copy number variations.

The dataset is essentially composed of four classes: *Organisms*, *Genes*, *Gene families* and the interactions between them. An example that shows the connection between different elements, to recover information about a gene in a particular organism, is shown in *Fig. 2.1*.



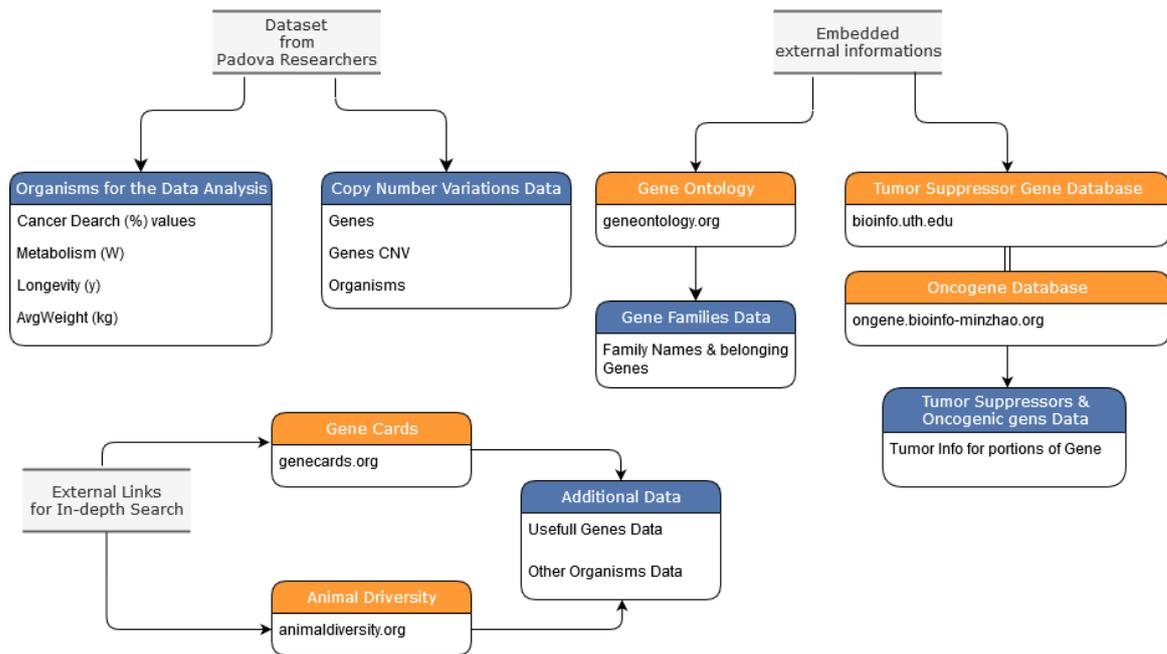
*Fig.2.1 Information flow representation for gene TP53 and organism LOXODONTA AFRICANA*

Most of the data has been provided by researchers from Padova, Chiara Vischioni and Cristian Taccioli.

They carried out a first stage of data collection for organisms and CNVs. The information correlated to known oncogenes, tumor suppressors and gene families have been recovered, according to the researchers, from collection of different genomic libraries available on-line.

Other data that is useful in the project come from the Gene Cards<sup>1</sup> and Animal Diversity<sup>2</sup> platforms. However, as it cannot be integrated for copyrights reasons, it has been included as external links for specific searches on genes and organisms.

The chart in *Fig.2.2* has the aim to show the origin of the different information that compose our dataset.



**Fig.2.2 Representation of the Dataset origin and composition**

<sup>1</sup> Genes Card - genecards.org

<sup>2</sup> Animal Diversity - animaldiversity.org

## 2.1.1 Organisms, Gene and Genes Family characteristics

The organisms that make up our dataset are split in two groups:

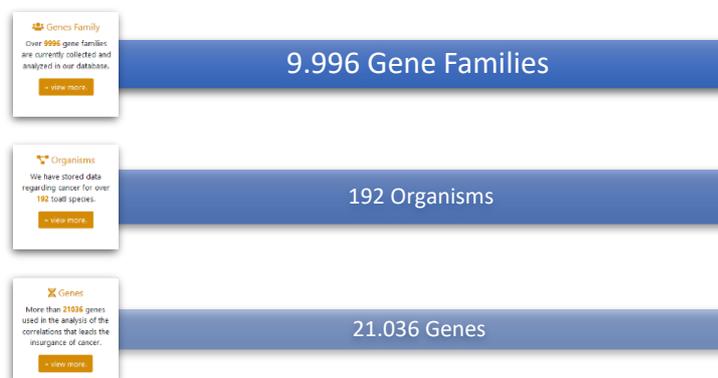
- Useful samples for the analysis [26]
- General Samples [192]

For all of them it's known the number of copies of each gene that builds up their genome. Only for the organisms useful for the analysis important parameters are known, such as cancer mortality and longevity. These samples are few due the difficulty to recover information regarding cancer rate of an organism. Sadly, many necropsies are needed to obtain this data especially when it comes from small samples.

The information in our disposal have been proved more than enough and has led to very good results. We can thus identify genes and families that distinguish cancer-resistant or extremely long-lived organisms. This offers the possibility to identify new valuable oncogenes and tumor suppressors towards which concentrate the research.

The platform contains information for a total of 21.036 genes with their CNVs values. For known tumor suppressors and oncogenic genes, this information is added as derived data in order to enable for a fast identification, specifically, within the platform are stored 1.217 tumor suppressors and 803 oncogenic genes.

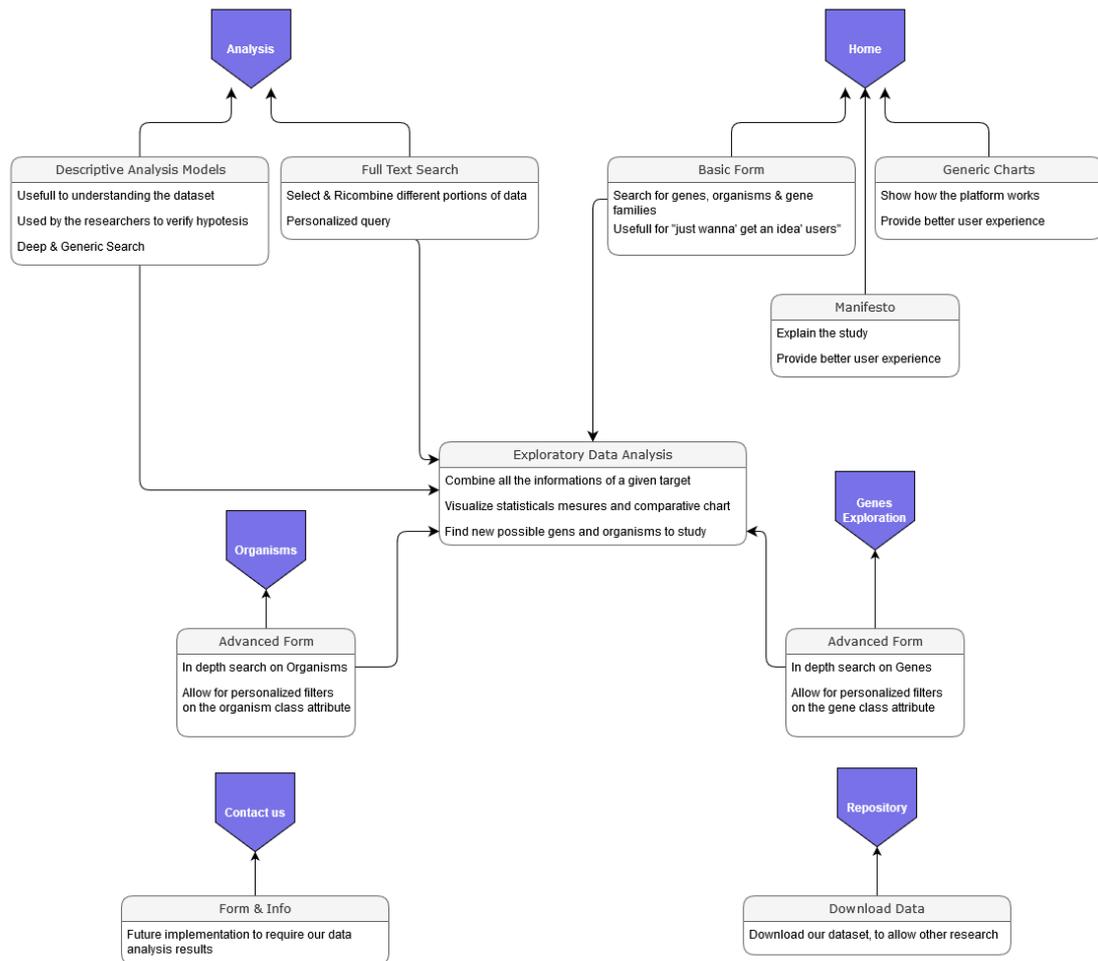
Another key element to understand the functions of a gene are the families to which it belongs. For the genes in our database, it is possible to trace the family information. We can define the belonging of a gene in as many as 9.996 different families.



*Fig. 2.3 Counters of genes, genes family and organisms - from the application Home page.*

## 2 A platform for researchers

What we've realized at the end is a Web Application made up by several components, in order to carry out different types of analysis on the data. The tools that allow specific actions inside each page are represented in *Fig. 2.4* which shows how the interactions between different component occurs.



*Fig. 2.4* A simple platform schema: Shows the interactions between areas of the platform and their components

### 3 Planning Choices

The realization of this complex web application for CNVs data analysis has been carried out by myself and my colleague Valentino Pisi. In the first phase, we worked together to implement an initial version of a Django web application and a database.

Each of us addressed different topics. The aspects I focused on are Exploratory Data Analysis technologies and various Descriptive Analysis Models with additional statistical measurements, specifically, I managed the user interaction with the platform, providing advanced search instruments to navigate the result and obtain dynamic reports.

My colleague Valentino Pisi worked on the management and manipulation of the data contained in the database, implementing a scalable data analysis strategy on it.

For more specific information on the aspects addressed by Valentino Pisi I encourage you to read his essay, entitled: “Scalable data science and technologies for the copy number variations studies”.

Based on the final objectives and functionalities that this application shall have, we identified the main components (libraries, coding languages and framework) that will help us, in order to build this platform.



**Django Web Framework** - *[djangoproject.com](http://djangoproject.com)*

Django is a high-level web framework, very powerful and useful for the creation of Python platforms and applications.

Many aspects related to the implementation of a web platform are automatically managed by Django, for example, it allows us to trivially manage requests to various resources, as the pages of the platform, and to interact with the database.



**FusionCharts** - [fusioncharts.com](http://fusioncharts.com)

Fusion Charts is a library for generating big data charts.

One of the most complex aspects of this project concerns the generation of appropriate reports, for each type of research done by the user, or, for a generic descriptive model. FusionCharts allows us to render charts for large amounts of data. Through advanced tools it is possible to dynamically manage events and data plots, this is essential for the Exploratory Data Analysis functions that need to give the user the possibility to visualize and explore different reports.



**Pandas** - [pandas.pydata.org](http://pandas.pydata.org)

Pandas is one of the most used tools for exploratory data analysis implementations and, in general, datasets manipulation of. It is used within the platform to recover statistical measures and manage the queries results.



**PostgreSql** - [postgresql.org](http://postgresql.org)

PostgreSQL is a powerful, open source object-relational database system. It has a strong reputation for feature robustness and performance.



**Scikit-Learn** - [scikit-learn.org/stable/](http://scikit-learn.org/stable/)

Is an useful tool for predictive data analysis, used within the application to run different statistical tests on the genes CNVs distribution for different groups of samples.

## 4 Descriptive Analysis Models

One essential information for the researchers that studies CNVs are the Descriptive Analysis Models and, therefore, they have been integrated in the platform. They allow the identification of new tumor suppressor or oncogenic genes and highlight trends of groups of samples known to have particular properties in terms of longevity and cancer rate.

To this end, the organisms were split into four different groups according to the two parameters of "cancer mortality (%)" and "longevity (y)".

[e.g. Organism Groups: based on cancer death (%)]

<b>“Cancer Resistant Organisms”</b> [Cancer $\leq$ 0.05%]	
<i>Organism</i>	<i>Cancer Death (%)</i>
<i>Heterocephalus glaber</i>	0
...	...
<i>Procvia Capensis</i>	0.0132
...	...
<i>Loxodonta Africana</i>	0.0311

<b>“Cancer Prone Organisms”</b> [Cancer $>$ 0.05%]	
<i>Organism</i>	<i>Cancer Death (%)</i>
<i>Tupaia Belangeri</i>	0.077
...	...
<i>Sus Scrofa</i>	0.28
...	...
<i>Rattus Norvegicus</i>	0.87

[e.g. Organism Groups: based on longevity (y)]

<b>“Long-live Organisms”</b> [Longevity $<$ 30y]	
<i>Organism</i>	<i>Longevity (y)</i>
<i>Homo Sapiens</i>	70
...	...
<i>Loxodonta Africana</i>	65
...	...
<i>Equus Caballus</i>	28

<b>“Non-Long-live Organisms”</b> [Longevity $\geq$ 30y%]	
<i>Organism</i>	<i>Longevity(y)</i>
<i>Mus Musculu</i>	1.5
...	...
<i>Canis Familiaris</i>	20.5
...	...
<i>Rattus Norvegicus</i>	2

Each model supports researchers in validating the correlation between CNVs of a gene and the occurrence of diseases related to cancer, or high longevity rates. They combine statistical measurements for genes CNVs values and organisms' parameters.

The user can interact with the models via a user interface that gives him/her various possibility to manipulate the shown results. The possible actions are:

- Selecting a new target group using a scroll-box that contains the four different groups of the analysis.
- Searching a gene, for an in-depth study, by writing the target in the predisposed input-box or by clicking on the desired gene in the chart.
- Visualize the result and by hovering the results in the chart obtaining additional information.
- Export the chart and the dataset in different format (CSV, XLSM sheet and more)
- 

Fig.4.1 shows all the possible interactions on a given model enabled to users.

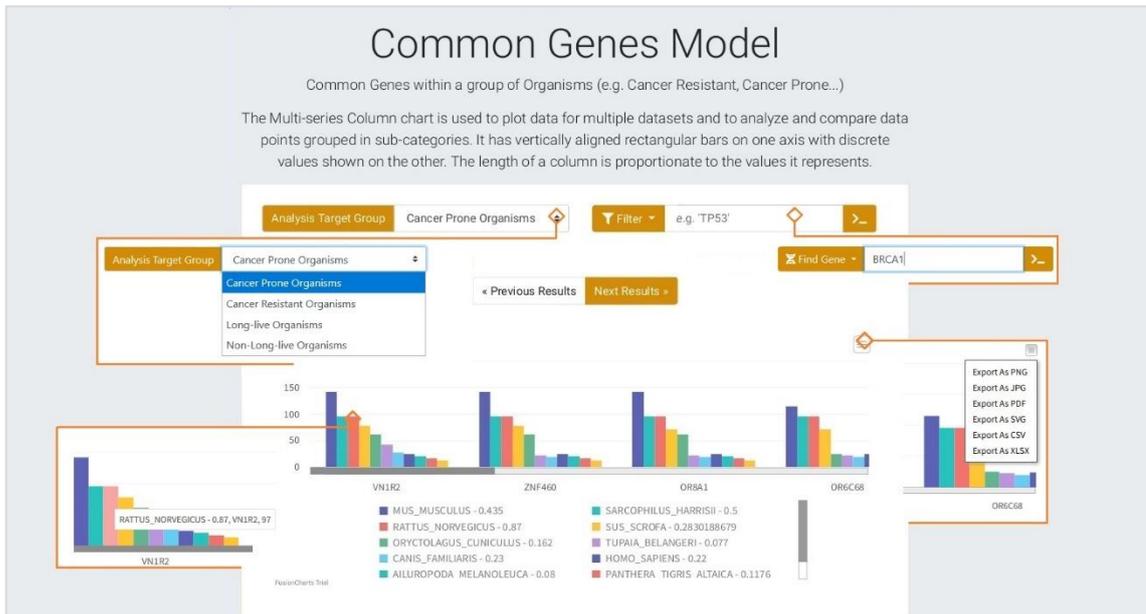


Fig. 4.1 Control panel & interaction with the “common genes” descriptive analysis model.

The aim of those models is also to obtain portions of data that are meaningful and easy to understand. It would be unthinkable to show a list of more than 21.000 unfiltered and randomly sorted genes, hoping that within them researchers could find relevant information.

A list of the Descriptive Analysis Models developed is written below.  
Each one will be deepened in the corresponding sections. (4.1,4.2,4.3)

*“Common Genes Model”:*

Highlights the CNVs of genes within the organisms of a group (e.g. cancer prone or long-live groups). The order of the results for each series is based on copy number decreasingly.

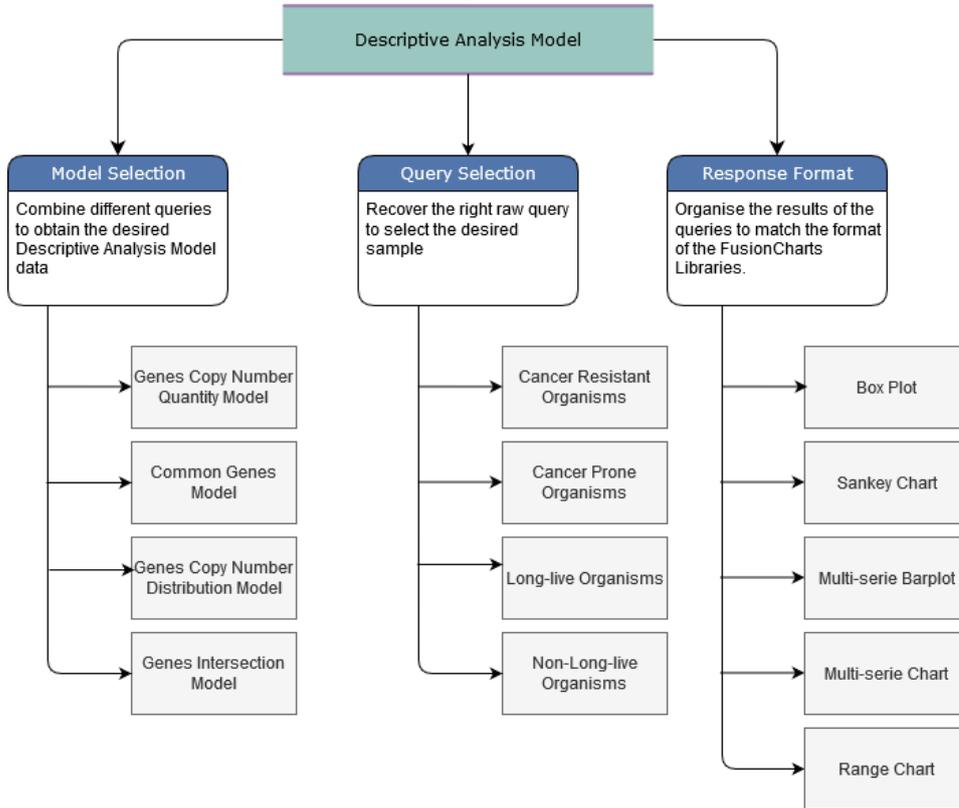
*“Genes Copy Number Quantity Model”:*

Useful to know which are the genes with highest copy number in the organisms of a group and how their statistical measures fluctuate. This model allows for a fast visualization of values such as mean, standard deviations, max and min. All the results are sorted by decreasing maximum number of copies of the genes.

*“Genes Intersection Model”:*

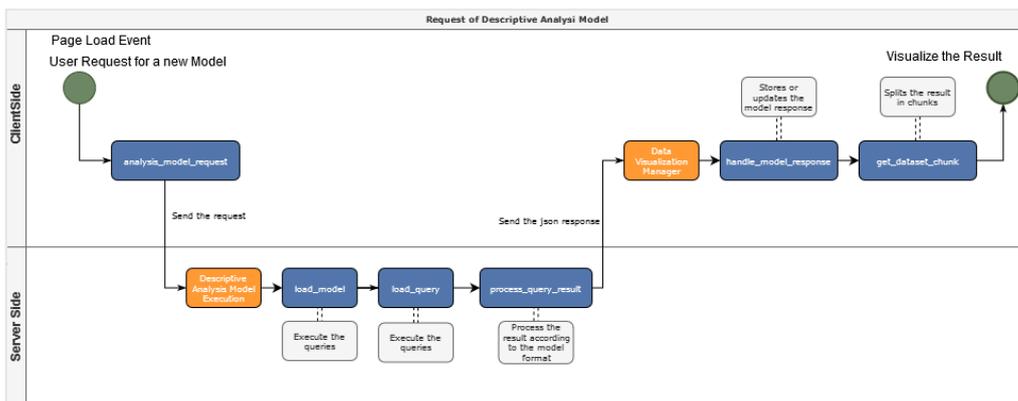
The differences between the copies of a gene for the samples of two different groups could highlight if a gene is interesting for the analysis. Furthermore, by showing the maximum number of copies in one of the samples within the two groups we can identify organism to study more in detail.

The generation of a generic Descriptive Analysis Model happens through different methods, classes and libraries. This task it's very complex to implement due to the need to combine different queries dynamically and choose the right type of visualization that better suits their results. The steps needed, server-side, in order to obtain a final result for a model can be grouped into three main phases as listed in *Fig.4.2*.



*Fig- 4.2 Representation of the different phase of a Descriptive Analysis Model generation in our platform*

The models are automatically initialized when the "Analysis" page is loaded or when the user performs a filtering action to change the group taken in consideration. The diagram in *Fig.4.3* explains the process behind the generation of a model.

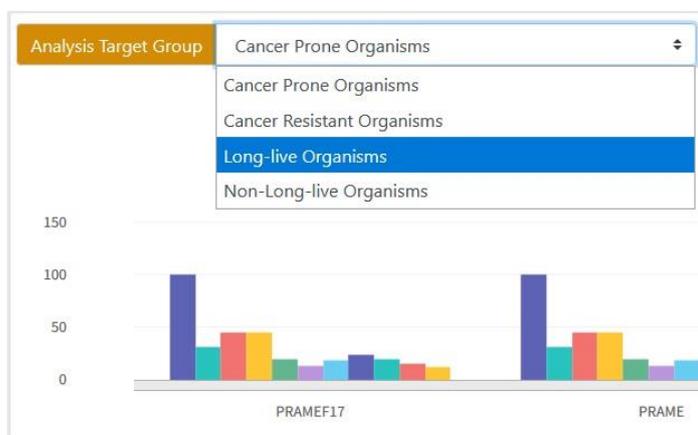


*Fig. 4.3 Schema for the generation of one single model, the request is triggered by the "Analysis" page load*

## 4.1 Common Genes Model

This model is based on the splitting of genes into different groups of organisms using cancer rate and longevity thresholds. It represents the CNVs of genes in a group selected by the user. It shows one main series of data, which are the common genes within a group, and for each of them their quantity (CN) in all the organism that belong to the selected group.

The order of the results (genes) is based on their copy number value decreasingly.



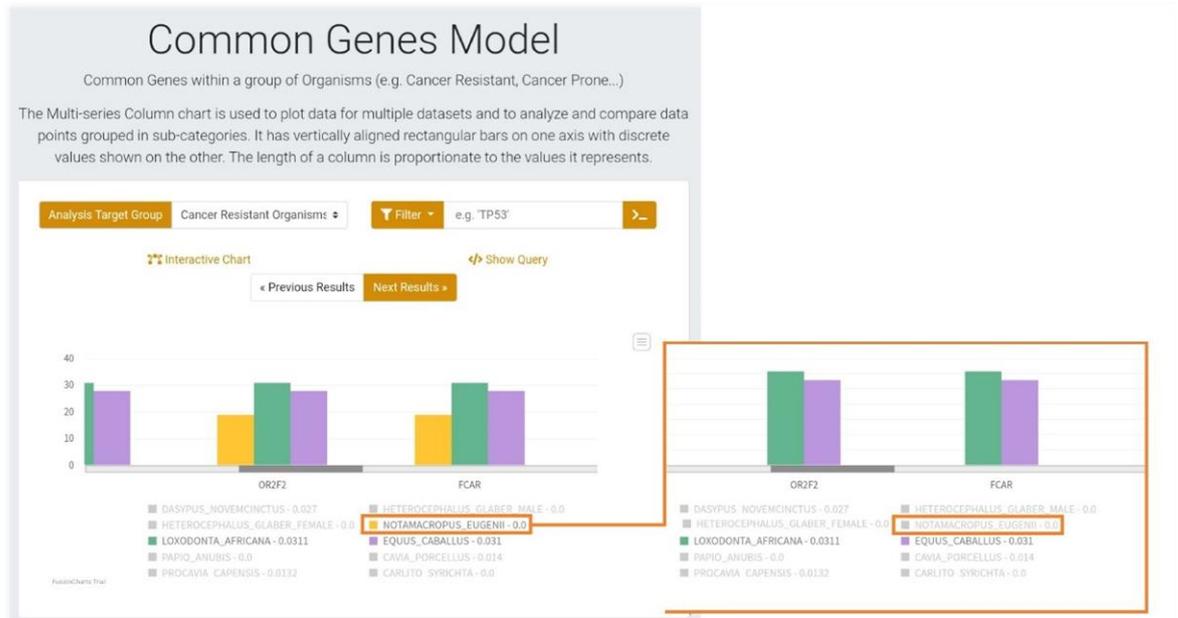
*Fig. 4.4 Analysis target Group scroll-box control.*

When the user selects a group, for example the "Cancer resistant organisms", the model will display all the genes common to the samples of this group.

To split the organisms into four groups we used a dictionary, in order to store the thresholds for the analysis. The format of the dictionary is 'threshold': 'threshold value'. For every query the splitting into groups can be easily done by filtering the desired threshold as a parameter. This little foresight allows to reuse the code with efficiency, as show in the code below.

```
# Execute the given raw query and returns it's result
def execute_query(self, query_id):
    query = eda_query_text(
        eda_target=self.eda_target, #Target of the EDA
        query_id=query_id,          #Required query identifier
        thresholds=self.thresholds #Thresholds values
    )
    #Opens a Conn with the db, execute the raw query and returns it's value
    c = db.connection.cursor()
    c.execute(query)
    query_result = c.fetchall()
    return query_result
```

Through this model the user has the possibility to exclude, or to include, samples from the analysis simply interacting with the chart as shown in *Fig.4.5*. It is thus easy to visualize and compare the quantities of gene in different organisms.



*Figure 4.5 Example of samples filtering in the common genes model – Removing Natomacropus Eugeni.*

The implementation of this model needs the combination of two different queries. What happens is that the data of the organisms are combined with genes copies, to provide a response that includes all the information. This operation is performed by the *load\_model* method and is shown in the following code.

```
def load_model(self):
    ... ..
    #Combine the query results for the given model
    #extends the result of the first query1 with the data returned by query2
    if self.selected_model == 'extend_model':
        for section in ['matching']: #We want that the queries results are about the same thing
            #We get the result of each query
            analysis_result = self.load_query(
                needed_venn_section=section,
                multiple_queries=True,
                sorting_field='copy_number'
            )
            #We process their results
            data = self.process_query_result(
                analysis_result,
                serie_name=self.selected_query[0]
            )
            #We need to handle possible duplicate in the response
            data['category'] = self.remove_duplicates_nodes(data['category'])
        ... ..
```

The *load\_model* method as shown in the picture above gets the raw mysql query code, that needs to be executed in order to obtain the desired result.

In this way, we have many “easy” pre-compiled queries that can be combined in different ways, allowing a good code reutilization, this also gave use the possibility to generate different results dynamically and include new analysis without the need to rewrite big portion of code.

The first query retrieves the genes belonging to the group, selected by the user, that have at least one copy in all the samples of the group.

```
# Common Genes Query: Common Genes of CancerNo, CancerYes, LongevityNo, LongevityYes
# Returns: gene_name, copy_number
if model_query == 'common_cancer_no_genes':
    query = f'''
        SELECT
            g.gene_name,
            MAX(cn.copy_number_qta),
            MIN(cn.copy_number_qta),
            ROUND(AVG(cn.copy_number_qta),3),
            SUM(cn.copy_number_qta),
            g.gene_dd_tumor_suppressor,
            g.gene_dd_tumor_oncogene
        FROM main_specie as s, main_gene as g, main_copyNumber as cn
        WHERE cn.copy_number_gene_id = g.id AND cn.copy_number_specie_id = s.id
        AND s.specie_cancer <= {thresholds['cancer']}
        GROUP BY g.gene_name, g.gene_dd_tumor_suppressor, g.gene_dd_tumor_oncogene
        HAVING COUNT(g.id) = (
            SELECT COUNT(s.id) FROM main_specie as s
            WHERE s.specie_cancer <= {thresholds['cancer']}
        )
        ORDER BY MAX(cn.copy_number_qta), SUM(cn.copy_number_qta), g.gene_name DESC
    '''
```

The following one instead returns parameters, such as cancer rate or longevity, of the organisms belonging to the group taken in analysis, therefore if a user select a group based on the cancer mortality this parameter is returned as an additional information.

```
# Generic Query: Genes of: CancerNo, CancerYes, LongevityNo, LongevityYes
# Recovers the information about the genes of copy number in the analysis species
# gene_name, copy_number, specie_name, specie_cancer/longevity
if model_query == 'specie_genes_cancer_no_genes':
    query = f'''
        SELECT
            g.gene_name gene_name,
            cn.copy_number_qta copy_number,
            s.specie_name specie_name,
            s.specie_cancer
        FROM main_specie as s, main_gene as g, main_copyNumber as cn
        WHERE cn.copy_number_gene_id = g.id AND cn.copy_number_specie_id = s.id
        AND s.specie_cancer <= {thresholds['cancer']}
    '''
```

Fig.4.6 shows how the selection of a group of samples, determined by a different parameter (cancer rate or longevity) causes the visualization of the respective additional data in the chart.

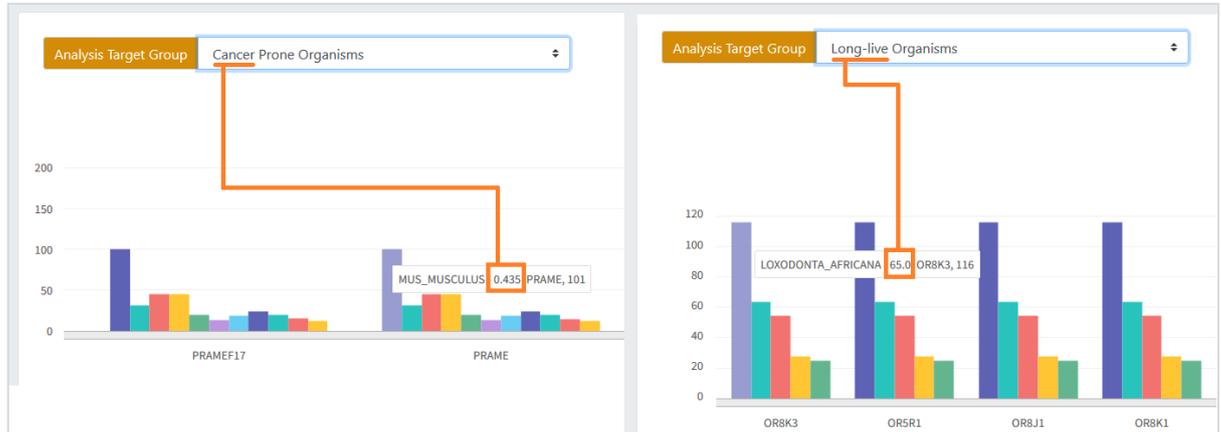


Figure 4.6 Additional parameters in the Common Gene Model as a result of the selected group.

The results of the queries then need to be organized in an appropriate format that satisfy the syntax of the graphical library (Fusion Charts) used for the generation of the interactive charts. The code listed below solve this task creating a dictionary that once converted into json-format will allow for fast chart rendering client-side.

```
# Response format for the generation of multiple-series column barplot
if self.response_format == 'categories_dataset_response':
    data['category'],data['dataset'] = [], []
    subseries = [] #Contains all the subseries labels
    subseries_data = {} #Contains the values of a given subserie
    series_parameters = len(analysis_result[0]) #Number of different values of a subserie
    for query_row in analysis_result: #We iterate over the rows of the query result
        if query_row[2] not in subseries: # If it's the first time we saw the serie
            data['category'].append({'label': query_row[0]}) #We add the main serie label
            subseries.append(query_row[2]) # Adds the label for the subserie
            subseries_data[query_row[2]] = { #Initialize the container of it's values
                'sub_value': [],
                'opt_value': query_row[3] if series_parameters>2 else ''
            }
        # We store the subserie multiple values
        subseries_data[query_row[2]]['sub_value'].append(query_row[1])
    #Finally we append the data to the response
    #We add all the subseries with their data
    for subserie in subseries:
        data['dataset'].append({
            'seriesname': f"{subserie} - {subseries_data[subserie]['opt_value']}",
            'data': [({'value': value}) for value in subseries_data[subserie]['sub_value']]
        })
```

The Common Genes Model to visualize the data uses a multi-series bar-chart. This data representation compares values and analyze them when their grouped in sub-categories. In our case, we look at different copy number values of a gene across the organism of a group. To observe this model with all its possible type of interaction refer to *Fig. 4.5, Chapter 4.*



Even though this model is the simplest one (as it doesn't need to combine different queries) is the one that highlighted the need to find an alternative way to handle the returned data, due to high number of results.

The executed query has the purpose of returning the distribution of the number of copies of a gene for the group selected by the user. In this way, the five numbers for the values of each gene can be calculated directly from the graphics library Fusion charts (once the distribution has been provided) at the time the chart is generated.

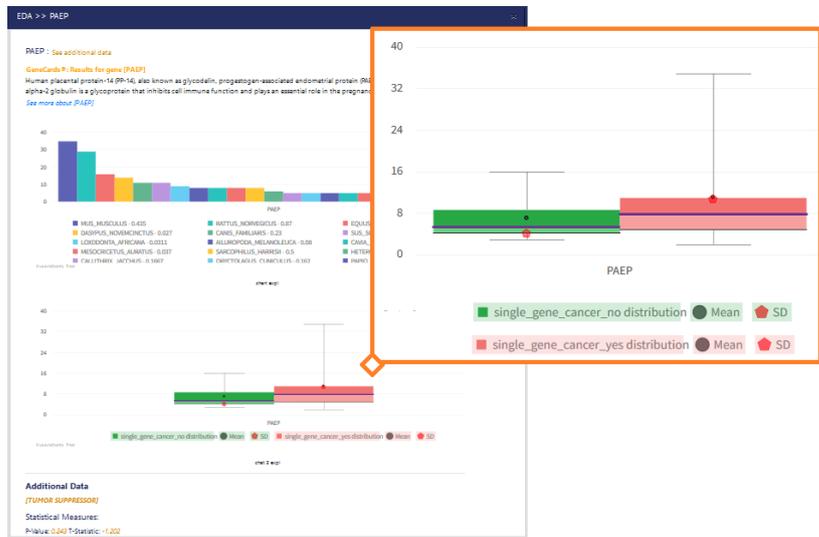
```

if query_id == 'all_cancer_no_genes':
    query = f'''
        SELECT
            g.gene_name gene_name, cn.copy_number_qta copy_number
        FROM main_specie as s, main_gene as g, main_copyNumber as cn
        WHERE cn.copy_number_gene_id = g.id AND cn.copy_number_specie_id = s.id
        AND s.specie_cancer <= {thresholds['cancer']}
        ORDER BY cn.copy_number_qta DESC
    '''

```

The type of chart used to represent this result is a statistical chart called boxplot. This way to visualize data is commonly used to examine and summarize a range of data values. It draws a statistical conclusion using the five-number summary.

This model is used as part of the Exploratory Data Analysis response. When the users ask for an in-depth search, on a single gene, a report with the result of this model is added, that shows the differences in the copy number distributions of a gene for both cancer resistant and cancer prone organism, as shown in *Fig.4.9*.



**Figure 4.9** A look at the Exploratory Data Analysis Report: how the Genes Quantity Model is incorporated.

### 4.3 Genes Intersection Model

If it is important to observe the tendencies that the number of copies of a gene has within a group, it is also legitimate to ask how this value differs for different groups.

A tumor suppressor gene should tend to have multiple copies in cancer resistant organisms. On the contrary, we expect to find oncogenes in larger copies within the genome of cancer prone species.

To be able to compare the maximum number of copies of a gene for organisms of two different groups may allow to identify new genes or anomalies to be further explored.

For this reason, the model "Genes Intersection between groups" has been implemented. The phenomenon to be observed is linked to the variation in the maximum number of copies of a gene in samples of different groups selected by the user.

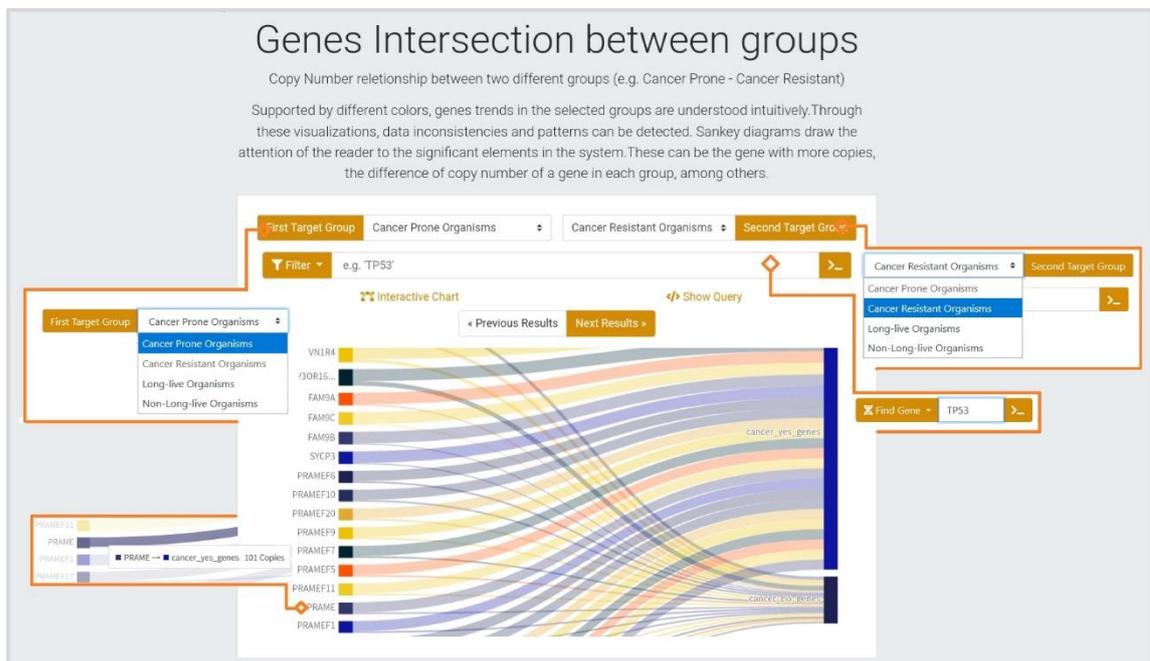
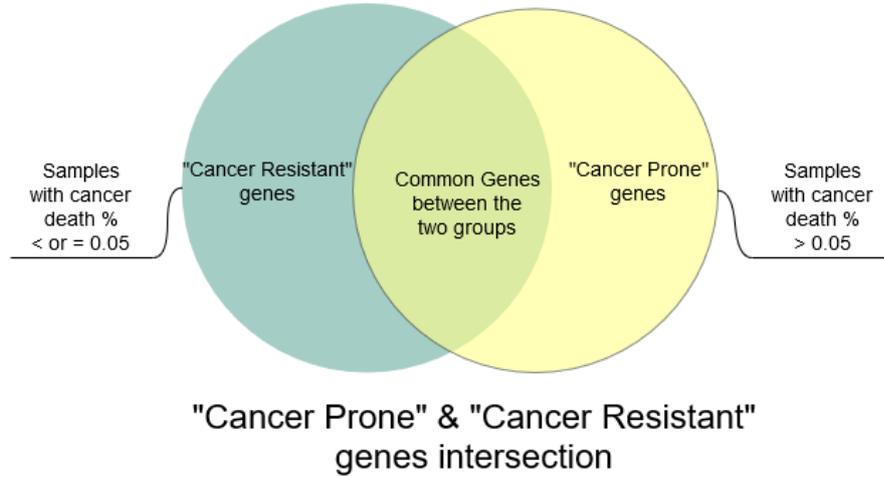


Fig. 4.10 A schema of the Descriptive Analysis Model “Genes Intersection Between group”

To build this model we need to combine multiple queries, the purpose is to get the three sections of a Venn diagram, that describe the genes of the two groups selected for analysis, as shown in *Fig.4.11*.



*Fig. 4.11 Venn Diagram to explain the result obtained with this model.*

The raw query to retrieve the genes of the three sets of the figure above is done dynamically, since the groups can be combined differently from the user.

The purpose of the underlying code portion is to join the two sub-queries that retrieve the genes of the user-selected groups, to obtain the genes common to both. The same happens for the other sections.

```
# Intersection Query
# - Target Groups: CancerNo, CancerYes, LongevityNo, LongevityYes
# - Target Values: gene_name, copy_number_avg (=average copy numbers of a gene in the species of the selected group)
def get_query_text_intersection(query_id, thresholds, needed_venn_section, sorting_field, analysis_target):
    query = '' # Will contain the final raw query to be executed
    # We get the alias for the sub-queries from their query_id
    group_one_query, group_two_query = query_id[0], query_id[1]

    # We need to recover the common elements of two generic groups of samples
    if needed_venn_section == 'matching':
        query = f'''
            WITH {group_one_query} AS (
                -- We select the sub-query for the first Target Group of the analysis
                {get_query_text(group_one_query, thresholds, analysis_target)}
            )
            -- We search for the matching elements for those recovered by the query above
            SELECT * FROM {group_one_query} WHERE EXISTS(
                -- We select the sub-query for the second Target Group
                WITH {group_two_query} AS ({get_query_text(group_two_query, thresholds, analysis_target)})
                SELECT * FROM {group_two_query}
                -- The matching condition is that the gens are the same
                WHERE {group_one_query}.gene_name = {group_two_query}.gene_name
            )
            -- The matching results are ordered by the given sorting_field
            ORDER BY {group_one_query}.{sorting_field} DESC
            ...
        '''
```

The procedure is safe, as “queries id” are managed on the server-side.

This allows to reuse the code effectively even when other groups of organisms will be added in the future for the analysis purpose.

The result obtained after the execution of the queries is a list sorted by number of copies of the gens. It will be necessary however to group the genes by their name, to respect the format of the library Fusion Charts and to visualize how the number differs for the two groups.

Genes	Copies
VN1R4	143
...	...
VN1R4	31

Before

After

The last step is therefore to reorganize the results in order to realize a dataset from which the result of the model will be created on client-side.

```
# To create a Node - Links Sankey Chart
# This allow to rapresent the result of the Venn Diagram
if self.response_format == 'nodes_links_response':
    data['nodes'], data['links'] = [], []
    data['nodes'].append({'label': serie_name}) #Add the Group
    for element in analysis_result: # Add the Links to the group for each Node
        data['links'].append({
            'from': element[0],          # Name of the Element
            'to': serie_name,           # Group of the Element
            'value': int(element[1]),    # Value of the Element
        })
    data['nodes'].append({'label': element[0]}) # Name of the Element
```

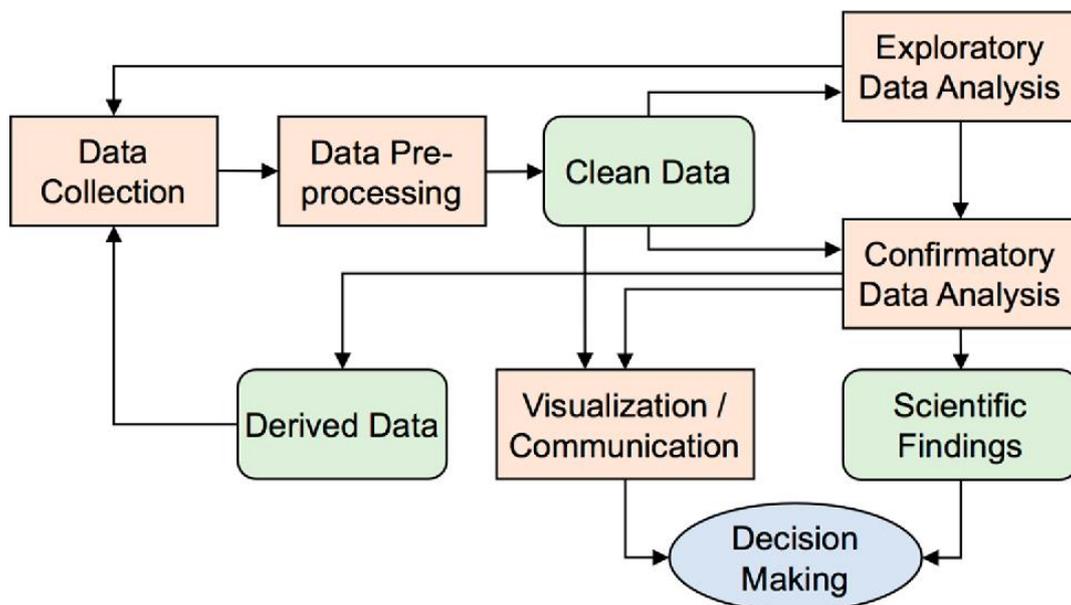
To see how this model looks and how the user can interact with it refer to *Fig.4.10* of this chapter (4.3).

## 5 Exploratory Data Analysis

In recent years more than ever there is the need to find new methods to visualize and manage big quantity of information. Many Data Engineering approaches tends towards the realization of descriptive analysis models, as those shown in the previous chapters, to observe tendencies and properties of a dataset which would be impossible to detect from raw data.

Descriptive models are the basis of the so-called Exploratory Data Analysis (EDA) tools, which combines their results in order to show accurate reports for the user, giving him the possibility to interact and explore datasets. We can say that EDA provides several tools, both conceptual and computational. Through this tool a researcher could discover patterns and develop new hypothesis, such as finding new TS or OG.

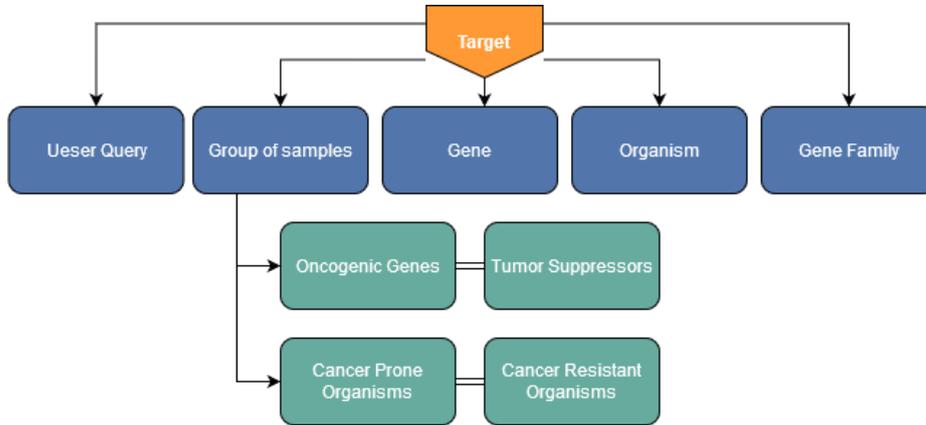
As introduced above, this tool helps significantly in the study of a phenomenon thanks to his property to test hypothesis extending the so-called confirmatory data analysis. EDA helps to interpret results of even well-specified theories and may lead to reveal anomalies or unexpected tendencies of the data. [8]



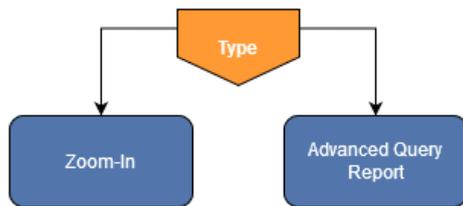
*Fig.5 Key steps in a data science process [9]*

To summarize, the role of EDA is to discover hidden properties and patterns within the data, to test hypothesis and check assumptions with the help of summary statistics and graphical representations and advance results interactions.

To implement EDA on our platform we selected two parameters: Type and Target. The "Target" identifies one of the classes that characterize our study and can be explored by the user to obtain complex report with the aim to describe them as well as possible. All the different targets are listed in *Fig.5.1*.



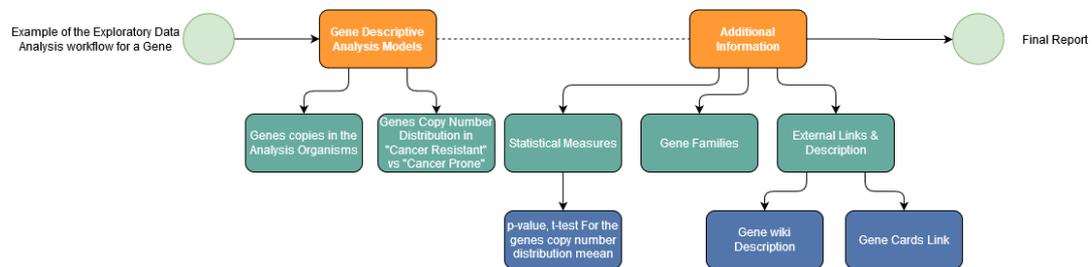
*Fig.5.1 Different targets of the exploratory data analysis in our platform*



*Fig. 5.2 Different action for a sample*

The "Type" parameter instead represent the action that can be performed on a given target.

When the user search for a particular sample to get it's information the "Type" parameter is set to "Zoom-In". The EDA module in this case retrieve all the information available on our database, organize them and return a detailed report to the researcher.



*Fig. 5.3 Exploratory Data Analysis process*

The diagram just shown explain how the Descriptive Analysis Models introduced in the previous chapter (4.1) are reused within the EDA, in addition to other general information, to generate a report.

## 5.1 Operation of the Exploratory Data Analysis

The request to perform the EDA can happen in different ways, the most common are: Through the search of a user via the forms (Analysis and Home pages) or caused by his interaction with a chart produced by one of the descriptive models.

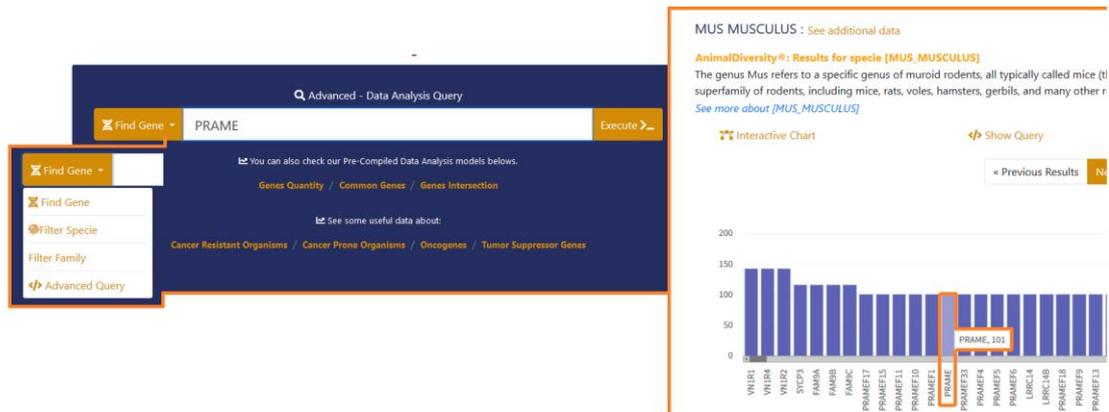


Fig. 5.4 Exploratory data analysis request

As a first result of a request the platform regain the descriptive analysis models (DAMs) results, to incorporate them into the final report.

In Fig. 5.5 the user has asked for a specific gene, therefore, two different DAMs are executed and their result is added to the response. As shown below.

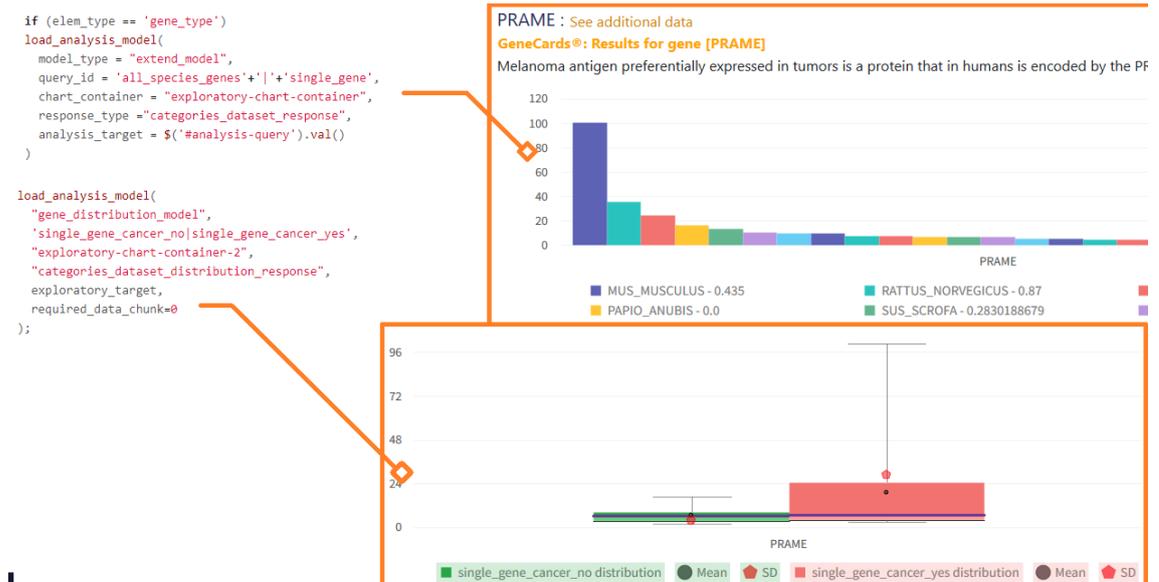
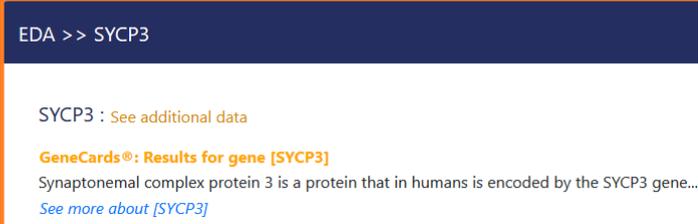


Fig. 5.5 Combining Descriptive analysis models for the EDA

The first part of the response includes a small description and useful external links to other platforms, this links change depending on the chosen target and allow to obtain a confirmation of the obtained result.

Fig 5.6 shows how this information is used within the report of a single gene.

```
try:
    wiki_link = wikipedia.page(str(self.eda_target) + aim).url
    wiki_summary = wikipedia.summary(str(self.eda_target) + aim, sentences=1) + "..."
except:
    wiki_link = '#'
    wiki_summary = 'Nothing Found'
# Preparing summary and sending data
summary = {'text': wiki_summary, 'link': wiki_link}
```



EDA >> SYCP3

SYCP3 : See additional data

**GeneCards®: Results for gene [SYCP3]**  
 Synaptonemal complex protein 3 is a protein that in humans is encoded by the SYCP3 gene...  
[See more about \[SYCP3\]](#)

**Fig 5.6 EDA Report: Searched sample description**

The next phase involves the collection of additional data obtained from the relationships between the searched sample and the data stored in our database.

The following portion of code (and report) show how different data related to a single gene, such as gene families and statistical measurements, are recovered and visualized.

```
# Recovers the info about a gene on the db
def add_gene_data(self):
    #Initialize the dictionary to store the information
    gene_data = {'gene_families': [], 'tumor_info': {}}

    #Recovers the informations about the gene families
    query_result = self.execute_query('get_gene_families')
    [gene_data['gene_families'].append(row[0]) for row in query_result]

    #Recovers if the gene is a tumor suppressor or an oncogenic gene
    gene_data['tumor_info'] = self.get_query_result('get_gene_tumor_info')

    #Some statistical measures about the gene copy number distributions
    ttest, pvalue = self.gene_statistical_measures()
    gene_data['statistical_measures']={'t_statistic': ttest,'p_value': pvalue}
```



**Additional Data**

Statistical Measures:  
 P-Value: 0.257 T-Statistic: -1.162

Related families:  
**ATP DEPENDENT CHROMATIN REMODELING**, CHROMOSOME ORGANIZATION, CELL CONFORMATION CHANGE, SEXUAL REPRODUCTION, ORGANELLE FISSION, MALE MEI IN REPRODUCTION IN MULTICELLULAR ORGANISM, SPERMATID NUCLEUS DIFFERENTIA REPRODUCTIVE PROCESS, GAMETE GENERATION, DEVELOPMENTAL PROCESS INVOLVE ORGANIZATION, PROTEIN DNA COMPLEX SUBUNIT ORGANIZATION, SPERMATID DIFFI, REPRODUCTION, MULTICELLULAR ORGANISM REPRODUCTION, DNA PACKAGING, CI CHROMOSOME, CHROMOSOME CENTROMERIC REGION, CHROMOSOMAL REGION, NI CHROMOSOME, MALE MEIOTIC NUCLEAR DIVISION, SPERM CHROMATIN CONDENSATI NUCLEOSOME ORGANIZATION, MEIOSIS I CELL CYCLE PROCESS, SYNAPTONEMAL STRL

[Download repo](#)

**Fig. 5.7 EDA Report: Searched sample additional data**

The statistical values shown in the Fig.5.7 and provided for a gene are the p-value and t-test, obtained with the “two tailed” Welch T-Test for unequal group of samples with different variance. [10]

With this test we look at the mean for the copies of a gene in two different groups of samples, in our case: "Cancer Resistant" and "Cancer Prone".

The hypothesis observed is that the average number of copies of the gene differs for the two population therefore the means aren't the same.

Statistical measurements like this are important to find new possible oncogenes or tumor suppressor. When a genes has a p-value lower or equal 0.05 we can reject the null hypothesis, this means in terms of our study that the gene discriminates well the groups and his high copy numbers values could have some relationship with low cancer rates.

The code below shows how the “scikit-learn” library is used to calculate the statistical measurements discussed before.

```
def gene_statistical_measures(self):
    #Execute the query to get the values of the distirbution of the searched gene in the cancer resistant group
    query_result = self.execute_query('single_gene_cancer_no_stat_values')
    mean1, std1, n1 = float(query_result[0][1]), float(query_result[0][2]), float(query_result[0][3])
    #Execute the query to get the values of the distirbution of the searched gene in the cancer prone group
    query_result = self.execute_query('single_gene_cancer_yes_stat_values')
    mean2, std2, n2 = float(query_result[0][1]), float(query_result[0][2]), float(query_result[0][3])
    # Calculate the whelch ttest (two tailed) different variance
    tstat, pvalue = ttest_ind_from_stats(mean1, std1, n1, mean2, std2, n2)
```

For an organism, instead, the additional data are represented by parameters such as: cancer mortality, longevity, metabolism and average weight.

The statistical information are simple measures of the number of copies of oncogenes and tumor suppressors within the chosen sample genome. This information allows the researchers for a better interpretation of the result as shown in *Fig. 5.8*.



**Fig 5.8 EDA Report: Organisms additional data visualized with range charts**

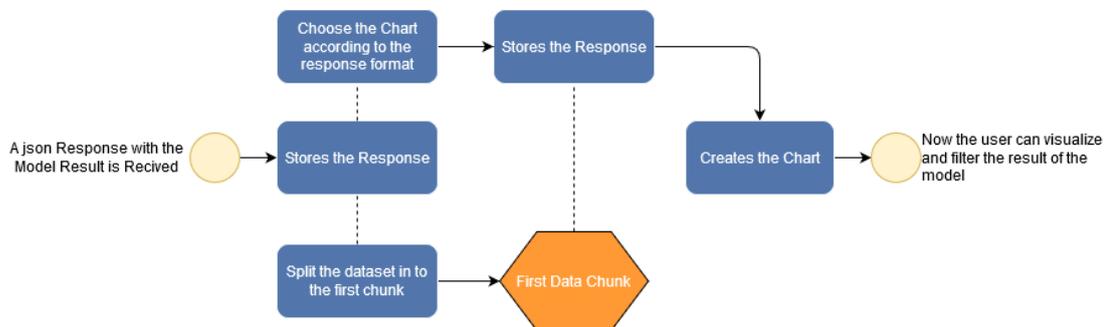
## 6 Data Visualization

The result obtained from a request of the user produce a high number of results, this is due to the number of returned genes for each organism, which is always close to 20.000 unit.

Trying to visualize those numerous results without first doing some pre-processing, results in un-efficient code and slow chart rendering, both direct cause of a bad user experience and platform functionality.

The solution to this problem is the division of the dataset (result of each DAMs) in multiple chunks.

This process is done client-side, indeed, even if the number of data is important, the exchange of information between client and server is not the critic aspect (the results data are well structured and are all strings or float values), what needs to be looked is therefore the chart rendering that happens locally.



*Fig. 6.1 Data visualization process*

200	POST	/start_analysis/	xhr	json	1,98 MB	1,98 MB	4246 ms
200	POST	/start_analysis/	xhr	json	1,25 MB	1,25 MB	1432 ms
200	POST	/start_analysis/	xhr	json	2,67 MB	2,67 MB	4778 ms
200	POST	/start_analysis/	xhr	json	2,39 MB	2,39 MB	4481 ms

*Fig. 6.2 Firefox developer console: Time to obtain the response of the Descriptive Analysis Models. (14Mbs)*

Once received the data are stored locally, by doing this is always possible to use them even in other moments without the need to send a new request to the server.

After this we split the data to obtain the first chunk and proceeds towards their visualization. Below is shown the function that handles the response of a generic model.

```

// This function receive the result of the Model from the Analysis & after saving the data locally
// calls the visualization module to render the response.
function handle_model_response(response, selected_model, chart_container, response_format){
  var data = response.model_data //Result of the analysis
  var chart_title = response.model_name
  //Check the size of the response to know if there was an error
  if(response.model_data.length == 0){=}
  else{
    $.each( stored_response, function( index, model ){=});
    if(old_response_index >= 0)stored_response.splice(old_response_index, 1 )
    //Update the new model result stored on th Client
    stored_response.push(=)
    //Choose the appropriate visualization model for the result
    data = get_dataset_chunk(selected_model, chart_container, response_format,require_chunk=0)
    draw_response_chart(data, chart_container, chart_title,response_format)
  }
}

```

For every type of response there is a format and a different method that operates to draw the desire chart. The different aspects to look at when we generate a chart are the parameters number and the relationship between data that we want to show

Examples of different data visualization techniques used in our platform can be:

- **Box Plot:** Commonly used to visualize the values of a distribution. This chart is used within the platform to visualize the distribution of the copies of a gene in two different groups.
- **Sankey Charts:** Represent the connection between two nodes. Is used as alternative to show the Venn diagram for the intersection of the gens for two groups
- **Multi-series line or bar plot:** Show the relationship between different series of data for a group of samples.

The Fig.6.3 shows the commonly used chart visualization for out study.

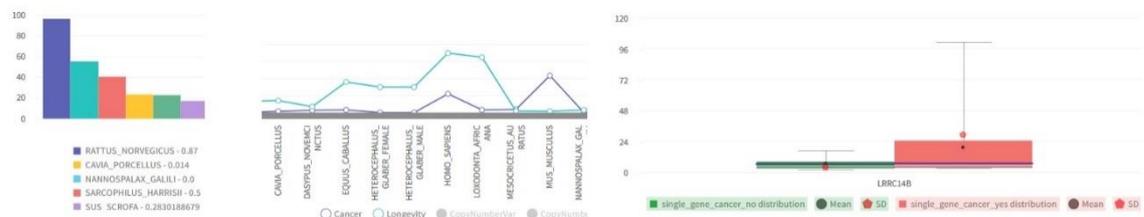


Figure 6.3 Different data visualization techniques

To give an idea on how this chart are generated below follows a portion of the jQuery script with the aim to create a bar plot from the dataset of result a model.

```
function draw_barplot(data, chart_container, data_title){
  var chartObj = new FusionCharts({
    type: 'scrollColumn2d', //Chart type
    renderAt: chart_container, //Container elem inside the page

    dataSource: {
      //Chart settings
      "chart": {
      },
      //Dataset for the chart
      "categories": [{ "category": data.labels }],
      "dataset": [{ "data": data.values }],
      "code": barplot_color,
    },
  },
}
```

Another thing worth noticing is the possibility to interact with the data, shown in a report. To implement this functionality in each chart, an event listener was added, it activates when the user clicks on a result sending a request for a search on the selected element.

```
events: {
  dataPlotClick: function(ev, props) {
    var exploratory_target = props.categoryLabel;
    load_analysis_model(=)
    load_eda(
      eda_query='None',
      eda_type='zoom_in',
      eda_target=exploratory_target,
      target_type='gene_type',
      selected_model = "extend_model",
      selected_format = "categories_dataset_response"
    )
  }
}
```

Fig.6.4 shows how a user after searching an organism and visualizing the gene with the highest copy number, decide to see additional data about that specific sample by clicking on it.

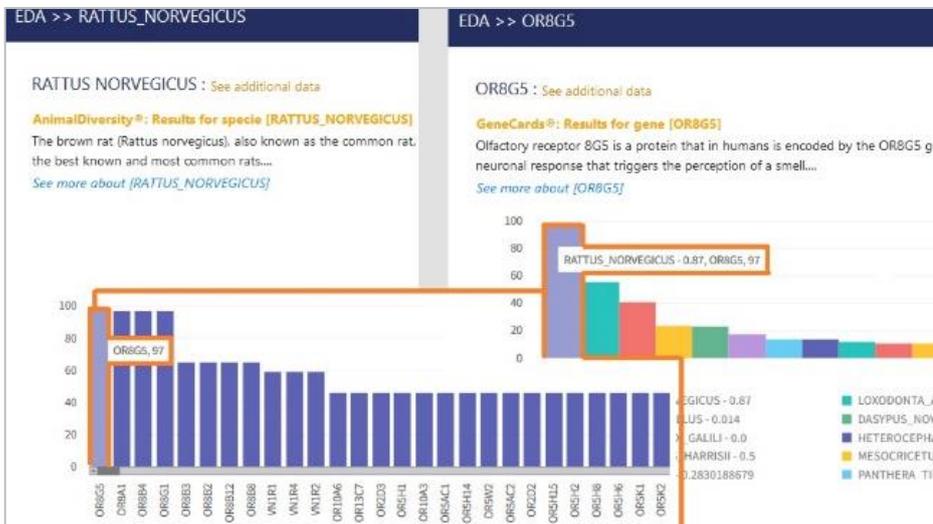
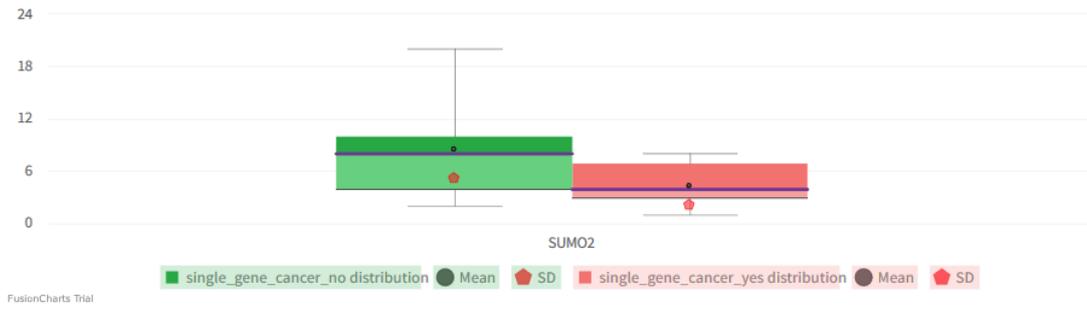


Fig. 6.4 navigation between different results of the EDA

## 7 Conclusions

Thanks to this platform researchers have already identified four different possible oncogenes and tumor suppressors, one of those samples is SUMO2 a gene never studied in depth, but that through the analysis on our platforms has shown having important characteristics for group of organisms with low cancer mortality rates.



### Additional Data

Statistical Measures:

P-Value: 0.023 T-Statistic: 2.455

*Fig.7 SUMO2 CNVs distributions for cancer-resistant and cancer-prone organisms.*

If this gene will have positive response to lab-testing than new drugs to attack or promote their functions could be produced, this is only an example of where this study can go and how our application can provide scientific utility.

This platform is indeed only born but it's potentials are evident in terms of possibilities for the study of scientific-biologic phenomenon, also, as an example of the technologies implemented as a solution to the analysis of different data in this field.

In a future several machine learnings tools could be added to this application, for the classification of genes, or, to look at the parameters that cooperates to promote (or slow down) the cancer onset.

# Bibliography

- [1] Elsevier, «Chapter 6 - Neoplasia,» *Pathology Illustrated (Seventh Edition)*, pp. 113-155, 2011.
- [2] V. N. G. J. V. & V. G. Andrei Seluanov, «Mechanisms of cancer resistance in long-lived mammals. *Nat Rev Cancer* 18,» 05 April 2018.
- [3] P. H. Dear, «Copy-number variation: the end of the human genome?,» *Trends in Biotechnology*, pp. 448-454, 1 July 2009..
- [4] A. D. a. J. P. d. Magalh, «Has gene duplication impacted the evolution of Eutherian longevity?,» *Aging Cell* , p. 978–980, 2016.
- [5] Viviane Callier, «Solving Peto’s Paradox to better understand cancer,» *PNAS*, p. 1825–1828, February 5, 2019.
- [6] A. M. B. a. C. C. M. Marc Tollis, «Peto’s Paradox: how has evolution solved the problem of cancer prevention?,» *Tollis et al. BMC Biology* , (2017) .
- [7] P. M. C. Carla Boccaccio, «Oncogeni e oncosoppressori,» *Enciclopedia della Scienza e della Tecnica*, 2007.
- [8] P. D. Record, «Principles and procedures of exploratory data analysis.,» *Psychological Methods*, 2(2), p. 131–160., 2016 APA.
- [9] D. H. M. B. M. a. o. Xiaogang Ma, «Using Visual Exploratory Data Analysis to Facilitate Collaboration and Hypothesis Generation in Cross-Disciplinary Research,» *Published in ISPRS Int. J. Geo-Information*, 2017.
- [10] G. D. Ruxton, «The unequal variance t-test is an underused alternative to Student’s t-test and the Mann–Whitney U test,» *Behavioral Ecology*, p. 688–690, 2006.
- [11] P. Patil, «exploratory-data-analysis,» 2018. [Online]. Available: <https://towardsdatascience.com/exploratory-data-analysis-8fc1cb20fd15>.



# Acknowledgements

Vorrei ringraziare tutti i professori e i compagni di lavoro che hanno consentito la realizzazione di questo progetto.

Ringrazio in particolare la Prof.ssa Federica Mandreoli, che ha reso possibile questa collaborazione con l'Università di Padova e ci ha prontamente affiancato nel corso dello sviluppo della piattaforma.

I ricercatori, Dott.ssa Chiara Vischioni e Prof. Cristian Taccioli, per averci dato la possibilità di collaborare realizzando un progetto a loro molto caro. Una nota particolare va aggiunta per la loro disponibilità e professionalità.

Ringrazio il Prof. Riccardo Martoglia e Prof. Luca La Rocca, per i preziosi consigli, rispettivamente in ambito informatico e statistico.

Non mancano certo parole di gratitudine per il Dott. Davide Ferrari che ha prontamente messo a nostra disposizione le risorse per ospitare la piattaforma, aiutandoci e supportandoci durante il nostro percorso.

Infine, ringrazio il mio collega e compagno di questo corso in Scienze Informatiche: Valentino Pisi. Senza rendercene conto abbiamo trascorso insieme quasi cinque mesi di stretta collaborazione per realizzare il nostro più grande progetto. Sono state numerose le difficoltà da affrontare, ma, avere un compagno su cui contare ha decisamente fatto la differenza.

Un pensiero speciale va ad amici e famigliari, per il supporto indiscriminato durante questo importante percorso.