Data Analysis Techniques & Managing solutions applied for the CNVs studies.

Elaborati di tesi:

- ➔ *Exploratory Data Analysis & Data Engineering techniques for the study of CNVs*
- ➔ *Scalable Data Science and Technologies for Copy Number Variations Studies*

Candidati:
Fabio Bove
Valentino Pisi

Relatore:
Prof.ssa Federica Mandreoli

Correlatori:
Prof. Cristian Taccioli
Dott.ssa Chiara Vischioni
Dott. Riccardo Martoglia

Università degli studi di Modena e Reggio Emilia

Università degli studi di Padova

Lo studio è nato dalla **collaborazione** fra l'**Università di Modena e Reggio Emilia** e l'**Università di Padova**, dalla proposta dei ricercatori Prof.Cristian Taccioli e Dott. Chiara Vischioni per **esaminare** il **genoma** di molteplici organismi al fine di ottenere informazioni utili alla **ricerca** di nuovi **oncogeni** ed **oncosoppressori**

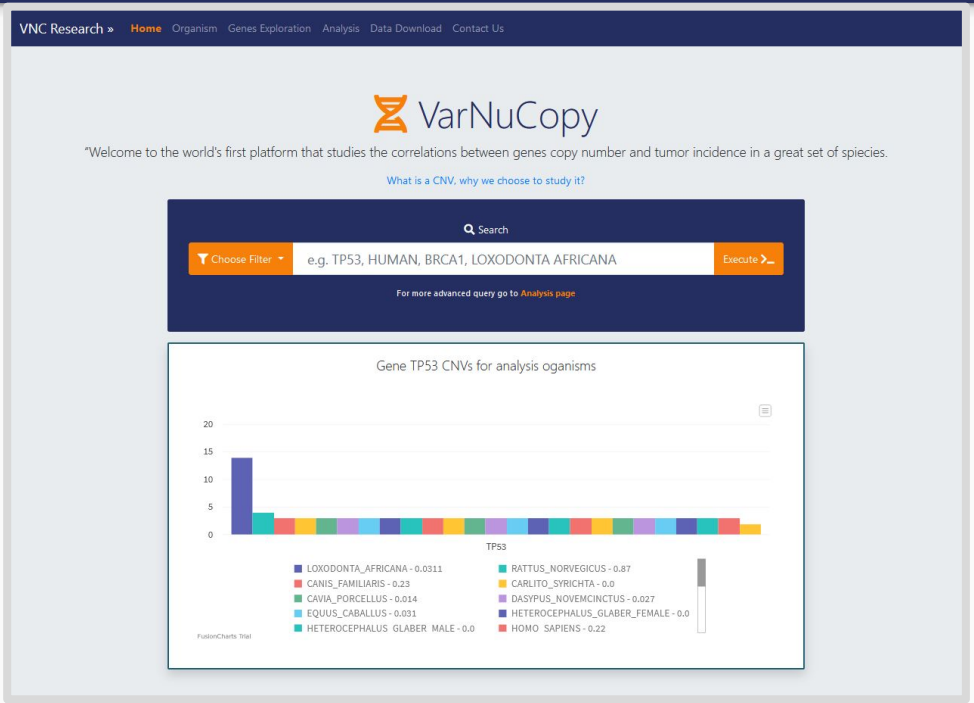
Di recente, la ricerca sul cancro ha preso in osservazione lo **studio** di **organismi** con particolari **proprietà** in termini di **resistenza al cancro** e con alti tassi di **longevità**, producendo importanti scoperte per **quelli** che sono i **meccanismi di resistenza** ed i **geni** a questi correlati

Lo studio di un fenomeno così complesso è dalle dinamiche ancora poco chiare, necessita la combinazione di differenti tecnologie e strategie a livello informatico che approfondiremo nei nostri elaborati

The study

Abbiamo quindi realizzato il primo **database** al mondo che andasse a combinare la **CNVs** di tutti gli organismi con i parametri biometrici fino ad ora considerati dai precedenti studi.

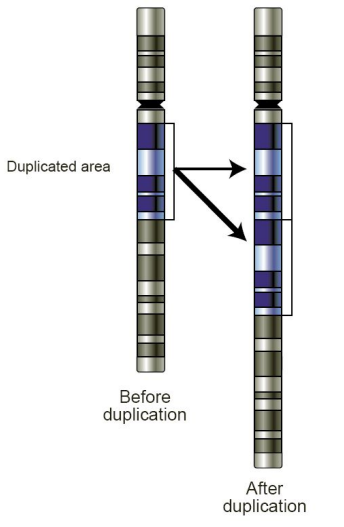
L'obiettivo finale è stato quello di creare una **piattaforma web** che incorpori tali strumenti e li renda disponibili e di facile accesso ai ricercatori.



Copy Number Variations

La **copy number variations** di un gene (**CNVs**) è quel fenomeno per il quale un gene si manifesta in maggiore, o minore, numero di copie all'interno del genoma di un individuo.

Queste **variazioni** di parti di genoma sono riconosciute come la **causa scatenante** di molte forme di cancro ed il loro studio si prefigge l'obiettivo di rivelare quali geni agiscono a tutela (**onco soppressori**) o a discapito (**oncogeni**) della salute dell'organismo.



Dataset

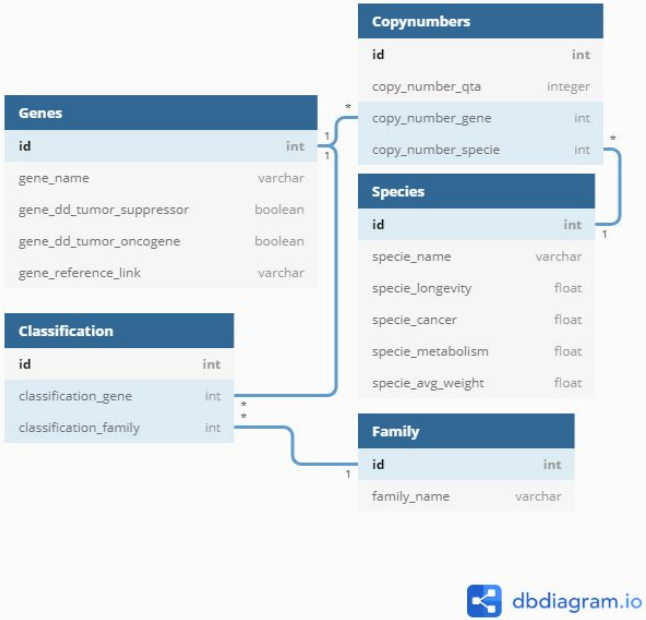
Abbiamo immagazzinato e recuperato:

- **famiglie geniche [9996]**
- **geni [21036]** di cui conosciuti:
  - **oncosoppressori [1217]**
  - **oncogeni [803]**

Altri dati quali i dati biometrici e del genoma degli **organismi [192]** ci sono stati forniti dai ricercatori dell'università di Padova.

In aggiunta sono stati inclusi link esterni a utili piattaforme di ambito biotecnologico-genomico per una verifica delle risposte generate:

- **Gene Cards**
- **Animal Diversity**



Ne è risultato un database composto da 5 tabelle e 7 milioni di tuple

Librerie - Tecnologie utilizzate

Per la realizzazione della piattaforma sono state adottate le seguenti tecnologie:

- **Django**
- **Pandas**
- **FusionCharts**
- **Scikit-Learn**

Architetture

Client-side tools

Exploratory Data Analysis: Implementa diversi **modelli di analisi descrittiva** misure e **test statistici**, offrendo ai ricercatori di osservare **pattern** e verificare **ipotesi**.

Realizza report personalizzati e ne permette **l'esplorazione** e **l'interazione.**

Server-side tools

Scalable Data Analysis: analisi di dati su fonti distribuite, anche a fronte di richieste onerose. Sono state implementate **strutture** e **meccanismi** che permettessero una risposta rapida ed efficace, da parte della piattaforma, con una breve analisi su questi.

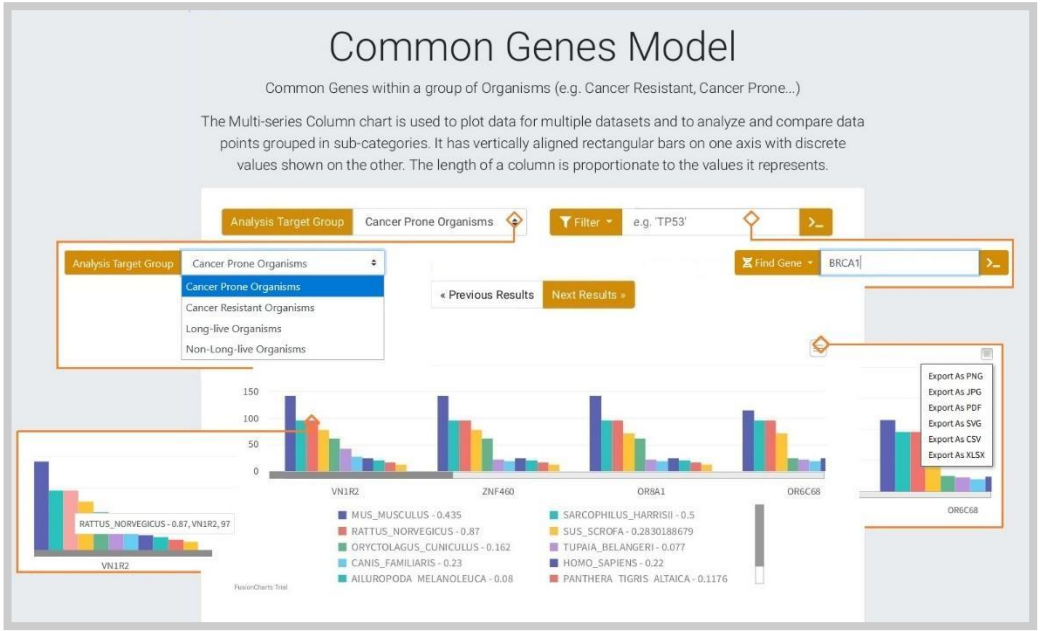
Exploratory Data Analysis & Data Engineering Techniques for the study of CNVs

Fabio Bove - A.A. 2019/2020
Università degli Studi di Modena e Reggio Emilia
Dipartimento di Scienze Fisiche, Matematiche e Informatiche

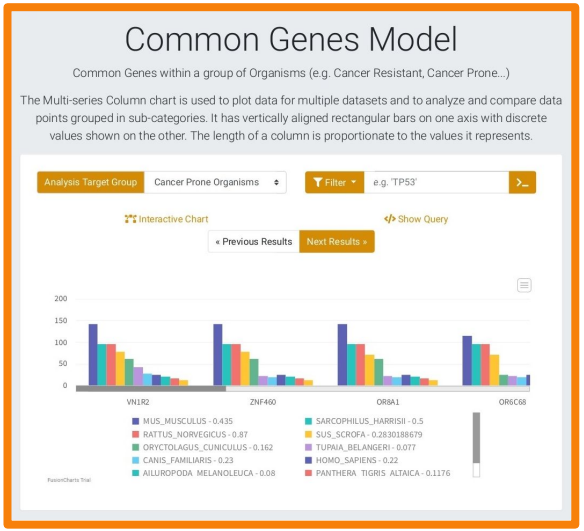
Technologies – Descriptive Analysis Models

Descriptive Analysis Models (**DAMs**):

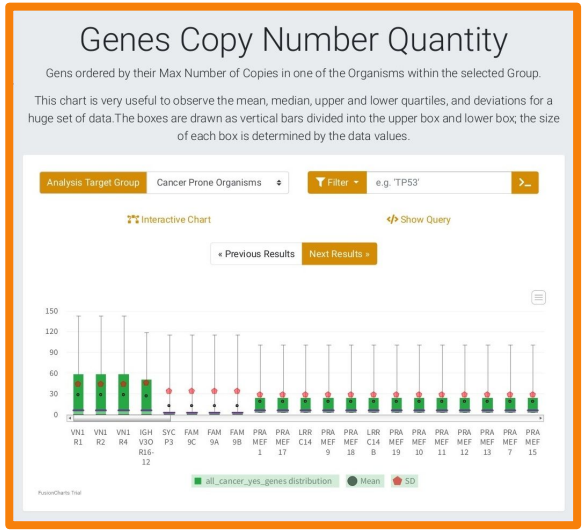
- Forniscono una **rappresentazione semplificata** di fenomeni del mondo reale.
- Recuperano **dati significativi** e li **organizzano** rendendoli comprensibili.
- **Evidenziano pattern** per campioni con particolari proprietà in termini di longevità e resistenza al cancro.
- Permettono di **identificare nuovi** possibili oncosoppressori ed oncogeni.



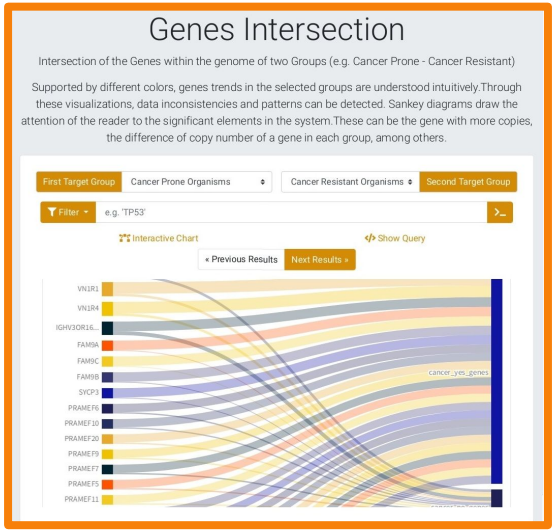
Descriptive Analysis Models



Evidenzia la **CNVs** di un **genes** all'interno degli organismi di un **gruppo.**
(e.g. cancer prone o long-live).



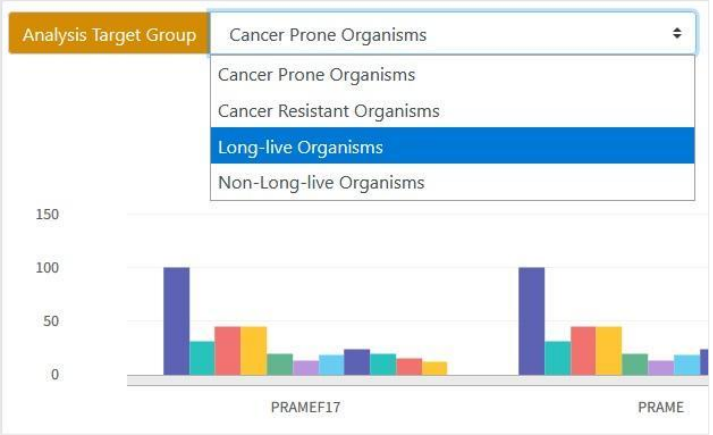
Utile per conoscere quali sono i **geni** con la **più alta CNVs** in uno degli organismi di un gruppo, sono incluse alcune **misure statistiche**(5 numeri).



Mostra **variazioni,** in termini di **massimo n°** di **copie,** di un **gene** per campioni di due **gruppi differenti.**

Descriptive Analysis Models – Gruppi analisi

Per implementare questi modelli, gli **organismi** sono stati suddivisi in quattro **gruppi differenti** secondo i parametri: **cancer mortality** e **longevity**.



Per **ogni query** la suddivisione in gruppi viene realizzata includendo la **soglia desiderata** . Questa accortezza permette di **riutilizzare** il **codice** con **efficienza**.

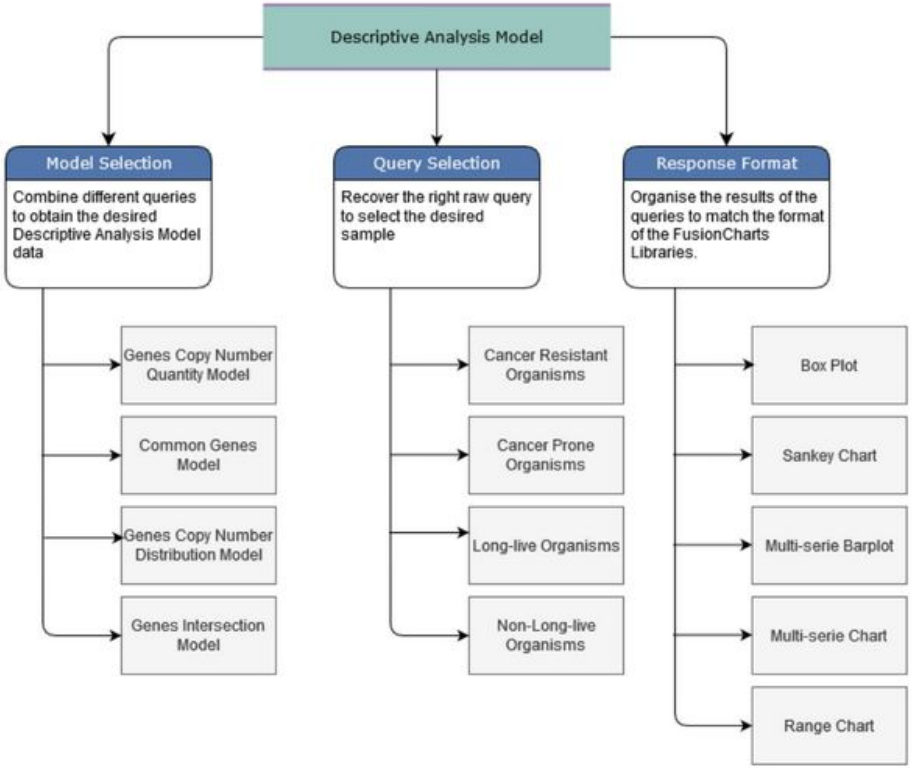
```
# Execute the given raw query and returns it's result
def execute_query(self, query_id):
    query = eda_query_text(
        eda_target=self.eda_target, #Target of the EDA
        query_id=query_id,          #Required query identifier
        thresholds=self.thresholds  #Thresholds values
    )
    #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
```

Descriptive Analysis Models - Generazione

Model Selection:
Combina le queries in maniera differente per ottenere il modello desiderato.

Query Selection:
Applica le soglie e recupera i dati del gruppo di campioni desiderato.

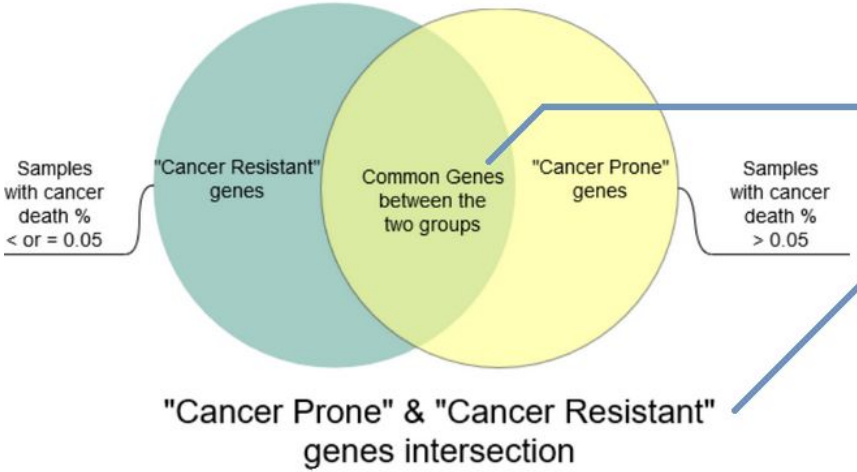
Formatting the Response:
Organizza i risultati per rispettare la sintassi della libreria grafica "FushionChart" per la data visualization.



Descriptive Analysis Models – Combinazione

Il codice mysql di ogni query viene composto **dinamicamente**, in quanto la selezione dei gruppi per l'analisi avviene secondo la scelta dell'utente.

La scelta è quella di avere sottoquery **query pre-compilate** combinate in base al modello ed ai gruppi selezionati




```
group_one_query, group_two_query = query_id[0], query_id[1]
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
```

Descriptive Analysis Models – Ordinamento

I **risultati ottenuti** dopo l'esecuzione delle query devono essere ordinati secondo un criterio di **ranking** deciso dai ricercatori.

Si utilizza il massimo numero di copie di un gene, come indice qualità e si raggruppano i campioni.

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

Before

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

After

L'ultima fase prevede di **riorganizzare** i **risultati** in per ottenere il dataset dal quale generare il grafico del modello.

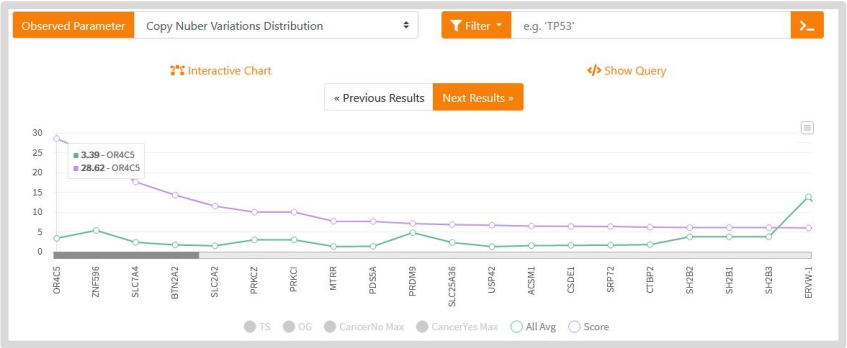
```
# To create a Node - Links Sankey Chart
# This allow to rapresent the result of the Venn Diagram
if self.response_format == 'nodes_links_response':
    data['nodes'], data['links'] = [], []
    data['nodes'].append({'label': serie_name}) #Add the Group
    for element in analysis_result: # Add the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

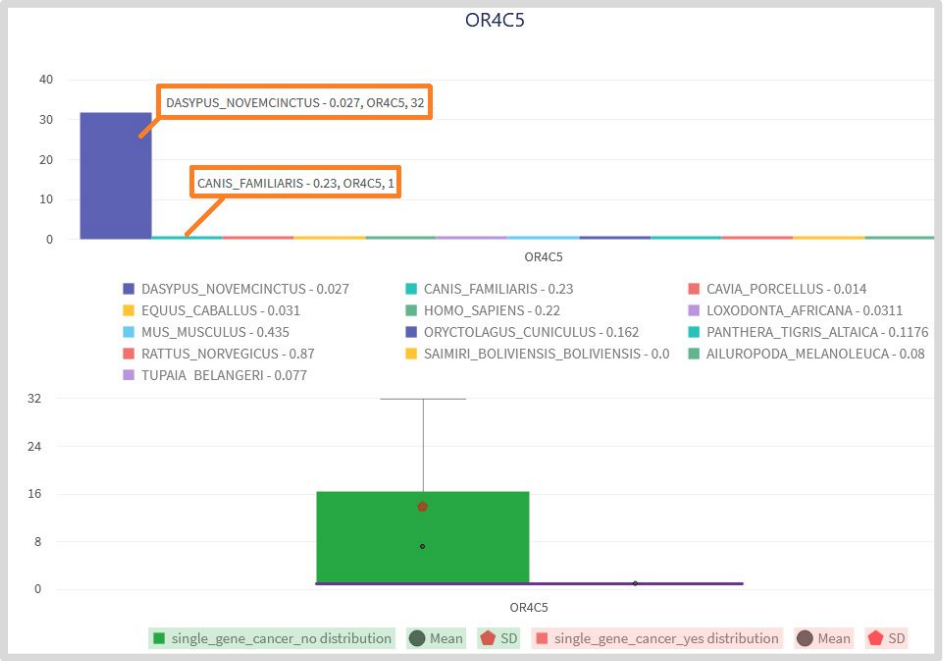
Descriptive Analysis Models – Ranking

Grazie al risultato ottenuto dall'incontro con il Prof. La Rocca, siamo riusciti ad individuare un **nuovo** meccanismo di **ranking** (modello "Genes Trend").

Privilegia geni con comportamento particolare:

- **elevate copie** per organismo **resistente al cancro**
- **distribuzione** del numero di **copie bassa** per il resto della **popolazione**





Technologies – Exploratory Data Analysis

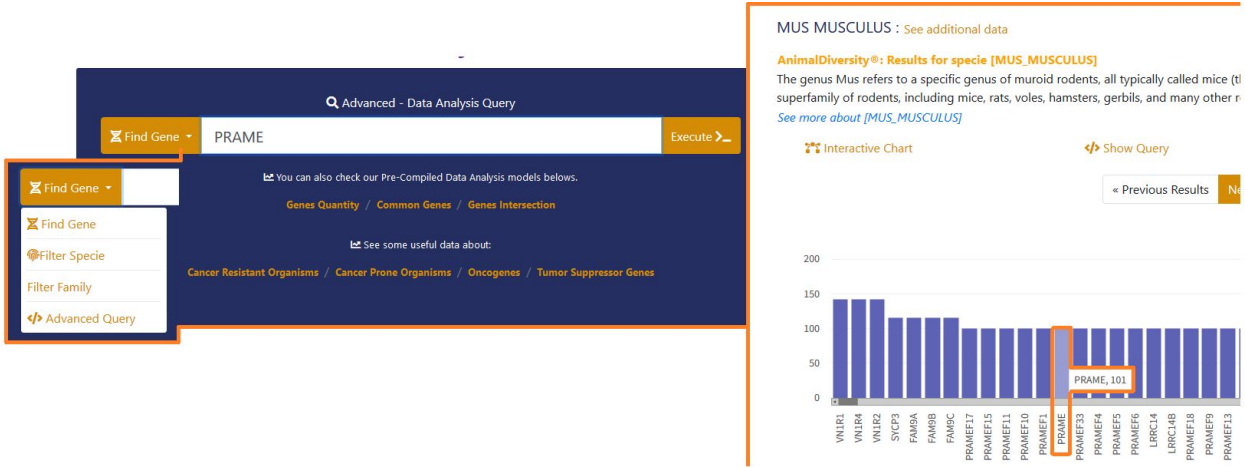
I diversi scopi dell’ Exploratory Data Analysis sono:

- **Testare ipotesi** e attraverso strumenti di visualizzazione e misure **statistiche**.
- **Combinare** differenti **DAMs** per mostrare report accurati all’utente.
- Fornire strumenti avanzati per **interagire** con i dati ed **esplorare il dataset**.

Exploratory Data Analysis – Richiesta

L' EDA viene reso disponibile all'utente secondo due principali strategie:

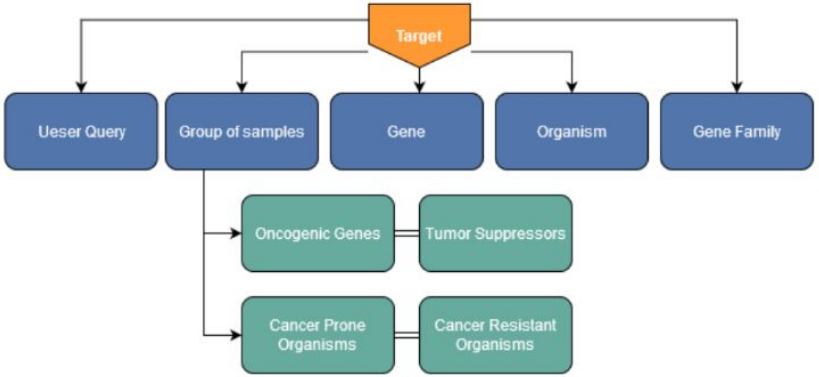
- Attraverso i **forms**, per la ricerca di un campione specifico
- **Interagendo** con un grafico, prodotto da uno dei modelli.



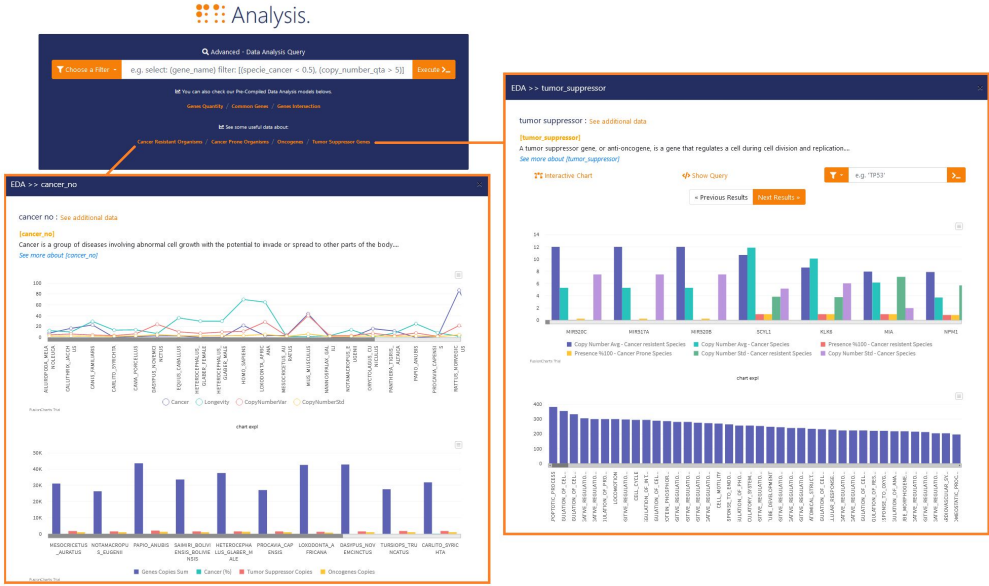
Exploratory Data Analysis – Target

L’**EDA** può essere applicato a **diversi target** a seconda delle necessità dell’utente.

Un **target** identifica una delle classi che caratterizza il nostro studio e può essere **esplorata** dall’utente.



Ogni **target** ha delle sue proprietà, per questo si ha la necessità di generare **report personalizzati.**



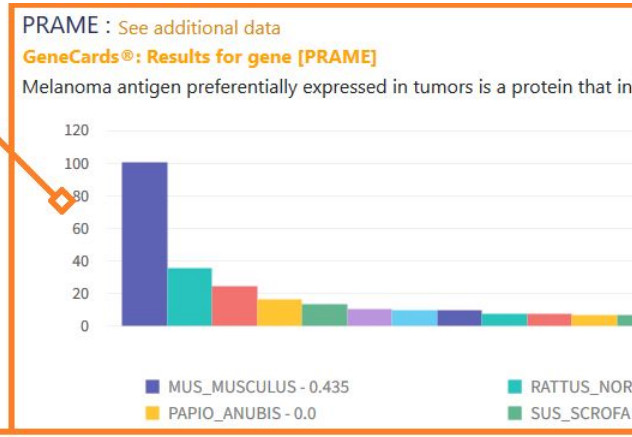
Exploratory Data Analysis - Combinazione DAMs

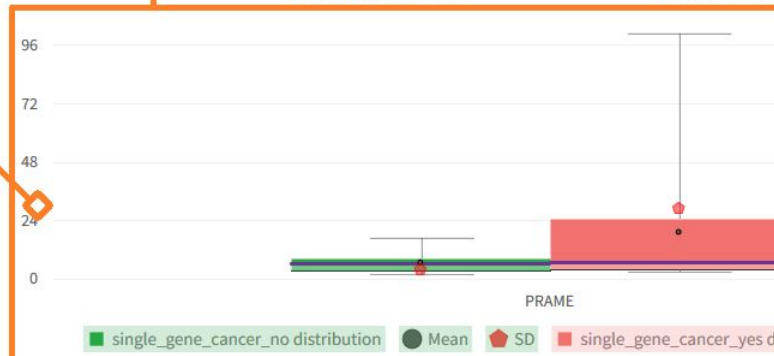
Il primo step, effettuato per realizzare il tool di Exploratory Data Analysis, consiste nel recuperare i risultati dei modelli descrittivi applicati al campione cercato.

```
if (elem_type == 'gene_type')
load_analysis_model(
  model_type = "extend_model",
  query_id = 'all_species_genes'+'|'+'single_gene',
  chart_container = "exploratory-chart-container",
  response_type ="categories_dataset_response",
  analysis_target = $('#analysis-query').val()
)
```

Nel caso di un singolo gene, ad esempio, i modelli osservano come la **distribuzione** delle sue **copie** varia negli organismi presi in analisi.

```
load_analysis_model(
  "gene_distribution_model",
  'single_gene_cancer_no|single_gene_cancer_yes',
  "exploratory-chart-container-2",
  "categories_dataset_distribution_response",
  exploratory_target,
  required_data_chunk=0
);
```





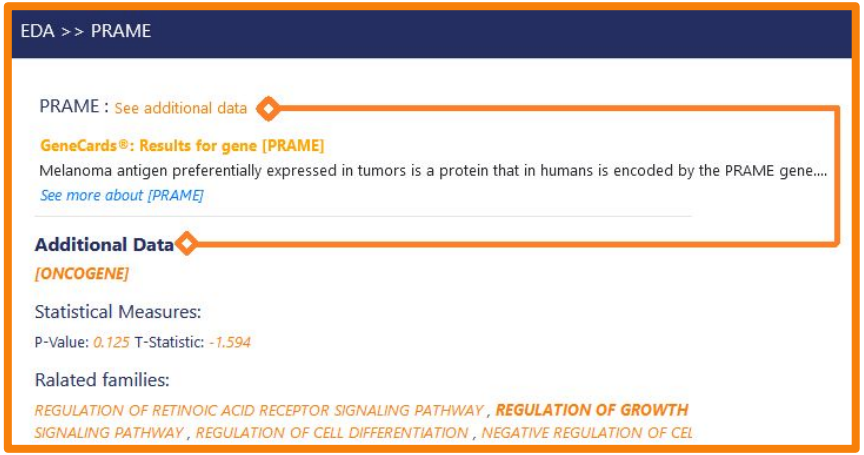
Exploratory Data Analysis – Dati aggiuntivi

Questa fase comprende la collezione di **dati aggiuntivi** ottenuti dalle **relazioni** fra il campione cercato e i dati memorizzati nel database.

```
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': pvalu
```

e.g. Se il campione cercato è un gene i dati aggiuntivi saranno:

- **Famiglie** geniche
- **Funzione** del gene (TS) (OG)
- **Test e misure statistiche** (p-value, t-test)
- **Descrizione** e **links esterni**

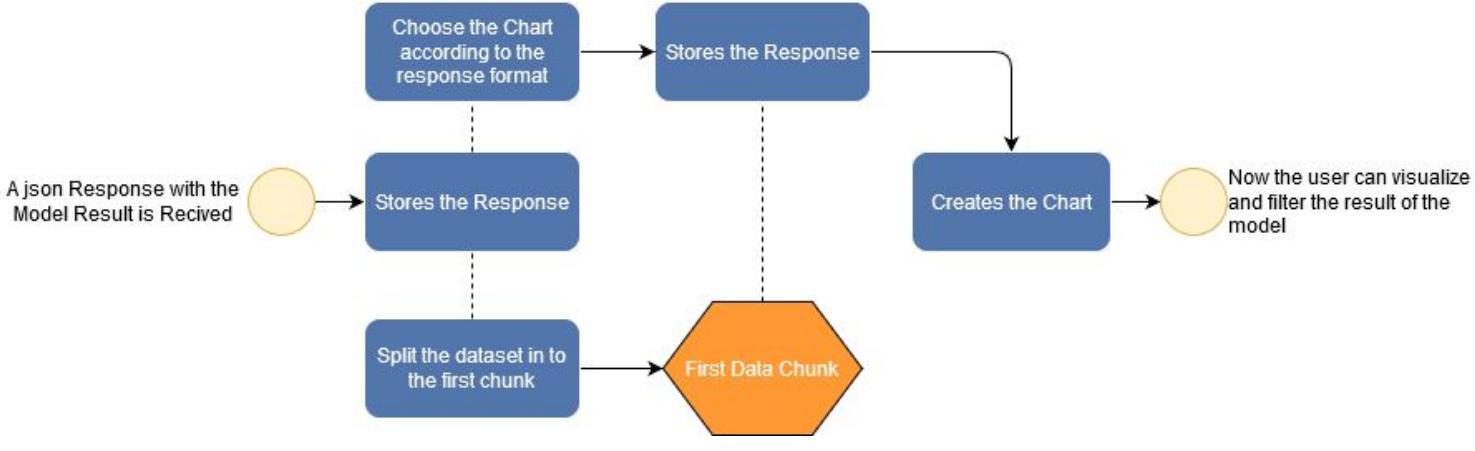


Technologies – Data Visualization

Le richieste dell'utente producono inevitabilmente un **alto numero di risultati**: sempre vicino alle 20.000 unità.

Provare a visualizzare questi dati senza **pre-processing**, comporta una inefficienza del codice e rallenta il **chart rendering**, risultando in una **cattiva user experience** e compromettendo le funzionalità della piattaforma.

Una delle soluzioni a questo problema è stata la suddivisione del dataset in **chunks**.

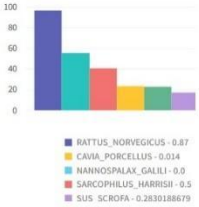


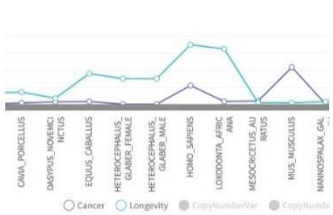
Data Visualization - Chunks & Rendering

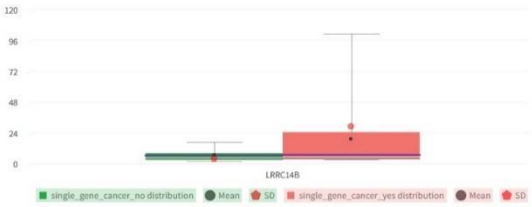
Una volta ottenuti i risultati sono **memorizzati localmente**, facendo questo è sempre possibile utilizzarli in un secondo momento, limitando così le richieste al server.

```
/// This function recive the result of the Model from the Analysis & after saving the data locally
/// calls the visualization module to rendeer 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 appropriete 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)
  }
```

Successivamente i dati vengono suddivisi per ottenere un **primo chunk** e procedere quindi alla sua **visualizzazione**.





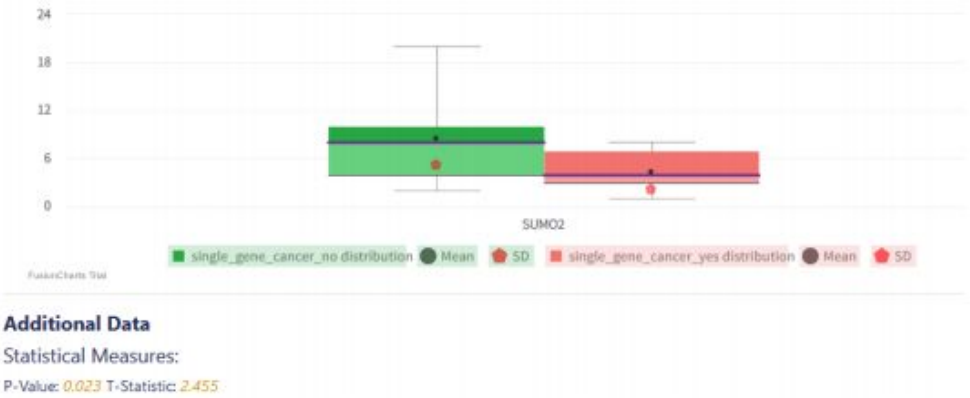


Conclusions

Grazie all’utilizzo di questa piattaforma i ricercatori hanno già **identificato almeno** quattro differenti possible **oncogeni** e **oncosoppressori**.

Uno di questi campioni è **SUMO2** un gene che attraverso l’analisi con gli strumenti da noi forniti ha dimostrato **importanti** caratteristiche per organismi con **bassa mortalità da cancro**.

Futuri sviluppi possono prevedere tools di **machine learning** per la **classificazione** di geni, oppure, per osservare i **parametri** che **cooperano** promuovendo o rallentando l’insorgenza del **cancro.**



Grazie per l’attenzione.