PROGETTO TRENITALIA

DATA MANAGEMENT AND VISUALIZATION

Anastasia Marzi
Master Degree | Data Science
University of Milano-Bicocca
a.marzi3@campus.unimib.it
Matricola: 847848

Alessandro Riboni
Master Degree | Data Science
University of Milano-Bicocca
a.riboni2@campus.unimib.it
Matricola: 847160

Federico Signoretta
Master Degree | Data Science
University of Milano-Bicocca
f.signoretta@campus.unimib.it
Matricola: 847343

19 giugno 2019

ABSTRACT

Considerata l'enorme quantità di persone che tutti i giorni utilizzano i mezzi pubblici per i propri spostamenti, i ritardi dei suddetti possono costituire un grosso problema.
Per svolgere il nostro progetto, abbiamo deciso di focalizzare la nostra attenzione sui ritardi compiuti dai treni sulla rete ferroviaria italiana. Grazie alle API del sito *ViaggiaTreno* di Trenitalia, siamo stati in grado di raccogliere dati nell'arco di un intero mese, quello di Gennaio.
Come prima cosa, abbiamo dovuto crearci un file che contenesse tutte le informazioni relative alle stazioni ferroviarie italiane integrando opportunamente e reperendo informazioni da diverse fonti.
In seguito, ogni sera nel periodo che va dall'1 al 31 di Gennaio, abbiamo raccolto i dati prodotti dai circa 8 mila treni che circolano in Italia ogni giorno.
Dopo aver ottenuto tutti i dati necessari, siamo passati ad una fase di pulizia, in cui abbiamo tenuto solamente le informazioni più rilevanti per la nostra analisi. A questo scopo, abbiamo sfruttato MongoDB, un DBMS (*DataBase Management System*) NoSQL basato sui documenti. Durante questo processo, abbiamo dovuto anche prestare molta attenzione alla gestione dei particolari e delle anomalie in quanto, in una prima analisi, ci siamo resi conto che i dati risultavano essere molto 'sporchi' e andavano quindi sistemati.
Dopo aver creato la nostra base di dati contenente tutti le informazioni necessarie, siamo passati alla parte di Data Visualization. Abbiamo creato quattro diverse infografiche attraverso le quali siamo stati in grado di mostrare il frutto del nostro studio.
La prima infografica è centrata sulla rappresentazione del traffico mensile e le altre tre sono volte a dare una panoramica sui ritardi: prima mostrando nel complesso cosa succede giornalmente in ogni provincia italiana e, in seguito, suddividendo lo studio fra treni regionali e dell'alta velocità.

***K*eywords** Treni · Ritardo · Italia · Regionali · Alta Velocità

Indice

Parte I
Data Management

1 Introduzione

L'Istat stima che nel 2017 circa 30 milioni di persone si siano spostate ogni giorno per raggiungere il luogo di studio o di lavoro: oltre un terzo della popolazione (il 35,5%) si sposta per motivi di lavoro, il 18,5% per motivi di studio.

Il pendolarismo, spiega l'istituto di statistica, riguarda oltre la metà della popolazione residente nelle regioni del Nord e nei grandi comuni; percentuali più basse si registrano nel Mezzogiorno e nei comuni di piccole dimensioni.

Per tutte le persone che tutti i giorni prendono il treno per i propri spostamenti, per andare a lavoro o a scuola il ritardo dei mezzi di trasporto e in particolare dei treni risulta essere un grosso problema. Per questo motivo, abbiamo deciso di raccogliere i dati dei treni italiani (quelli presenti su portale *ViaggiaTreno*) ed analizzarne il comportamento, prendendo come campione di studio l'intero mese di gennaio.

2 Data Ingestion e Storage

In questa prima fase del nostro progetto, ci siamo focalizzati sulla creazione della nostra base di dati. Le informazioni di cui avevamo bisogno erano quelle relative alle diverse stazioni italiane e a tutti i treni che passano giornalmente per quelle stazioni.
Come primo passo, quindi, descriviamo tutti i passaggi che abbiamo dovuto compiere per arrivare ad avere il dataset completo sulle stazioni.

2.1 Creazione del file contenente le informazioni delle stazioni italiane

Per avere un primo file di base che contenesse le informazioni su tutte le stazioni ferroviarie presenti in Italia abbiamo sfruttato le API del portale "ViaggiaTreno" di Trenitalia. Il procedimento è suddiviso in tre diversi step, descritti come segue:

Step 1: Abbiamo interrogato la base di dati del portale ViaggiaTreno per ottenere l'elenco delle stazioni italiane e il relativo codice identificativo, attraverso il seguente link:
http://www.viaggiatreno.it/viaggiatrenonew/resteasy/viaggiatreno/autocompletaStazione/A
Abbiamo dovuto ripetere questo procedimento per ogni lettera dell'alfabeto in quanto non era possibile scaricare tutti i dati in una volta sola.
La risposta che abbiamo ottenuto si presenta nella seguente forma:

```
ABANO TERME|S05700
ABBADIA LARIANA|S01416
ABBASANTA|S12873
ABBIATEGRASSO|S01062
ACATE|S12409
ACCIANO|S07423
ACERRA|S09215
ACIREALE|S12328
ACQUA ACETOSA|S08715
ACQUAFREDDA|S11722
ACQUAMELA|S09798
ACQUANEGRA CREMONESE|S01913
```

Step 2: Grazie ai codici identificativi delle stazioni ricavate durante il primo step, ci è stato possibile ricavare anche i codici identificativi associati alle diverse regioni italiane.
Per mezzo di questa coppia di codici siamo stati in grado di reperire informazioni aggiuntive sulle stazioni, in particolare le coordinate geografiche, quali latitudine e longitudine, di alcune di esse. Non è stato possibile individuare queste coordinate per tutte le stazioni italiane e per questo motivo abbiamo integrato e completato le coordinate mancanti utilizzando la libreria GeoPy di Python (la quale verrà presentata successivamente) e la libreria Wikipedia.

I dati ottenuti sono in formato json e si presentano come segue. Nell'esempio riportato, troviamo i dettagli relativi alla stazione di Torino Porta Nuova:


```
{"codReg":3,"tipoStazione":1,"dettZoomStaz":
[{"codiceStazione":"S00219","zoomStartRange":8,"zoomStopRange":9,"pinpointVisibile":tr
ue,"pinpointVisible":true,"labelVisibile":true,"labelVisible":true,"codiceRegione":nul
l},
{"codiceStazione":"S00219","zoomStartRange":10,"zoomStopRange":11,"pinpointVisibile":t
rue,"pinpointVisible":true,"labelVisibile":true,"labelVisible":true,"codiceRegione":nu
ll}],"pstaz":[],"mappaCitta":
{"urlImagePinpoint":"","urlImageBaloon":""},"codiceStazione":"S00219","codStazione":"S
00219","lat":45.060969,"lon":7.67747,"latMappaCitta":0.0,"lonMappaCitta":0.0,"localita
":{"nomeLungo":"TORINO P.NUOVA","nomeBreve":"Torino
P.Nuova","label":"Torino","id":"S00219"},"esterno":false,"offsetX":35,"offsetY":20,"no
meCitta":"Torino"}
```


Step 3: Come ultimo step, abbiamo immagazzinato il tutto all'interno di un primo file in formato csv.

2.2 Integrazione del file .csv sulle stazioni italiane

Il file csv che abbiamo creato nello *Step 3* del paragrafo precedente non era completo. Al contrario, esso era popolato da numerosi valori nulli e presentava più campi di quelli di cui avevamo bisogno.
Per questi motivi abbiamo dovuto effettuare un processo di *Data Integration* in cui abbiamo unito le informazioni ricavate da fonti diverse. Il nostro scopo è quello di ottenere un file csv finale che contenga solamente i seguenti attributi per ogni stazione italiana: **nome della stazione**, **provincia**, **regione**, **latitudine** e **longitudine**.
Dato che non è stato possibile ricavare le informazioni tutte dalla stessa fonte abbiamo dovuto servirci di diverse fonti esterne e di diversi strumenti per fare in modo che il dataset risultasse completo e senza dati mancanti. Di seguito, sono esposti i diversi passaggi che abbiamo dovuto compiere per svolgere questo lavoro.

2.2.1 Data Integration su Provincia e Regione

In primis, abbiamo avuto bisogno di collezionare le informazioni relative alla provincia e alla regione in cui risultano essere situate le nostre stazioni. Per fare questo, abbiamo inizialmente fatto *web scraping* da *trenitalia.it*: in particolare, da questo sito abbiamo ottenuto il comune e la provincia della singola stazione. Con questo procedimento abbiamo ottenuto un file csv in cui erano presenti tre campi: **Stazione**, **comune**, **provincia**. L'integrazione è stata fatta effettuando un join fra questo dataset e quello creato precedentemente, collegando i campi "Stazione" e "Nome Stazione".

A questo punto ci siamo serviti dei dati provenienti dall'*ISTAT* e da *OpenStreetMap* per arrichire il nostro dataset e per eliminare i valori nulli che ancora avevamo. Questo è stato possibile sfruttando i campi "comune" e "nome stazione" ricavati in precedenza.

2.2.2 Integrazione su Latitudine e Longitudine

Una volta integrati i campi categorici "provincia" e "regione", abbiamo sfruttato *Wikipedia* e *GeoPy* per integrare le latitudini e le longitudini mancanti. Di seguito vengono brevemente descritti i codici utilizzati per svolgere tale integrazione.

- ***GeoPy***: grazie a questa libreria di Python, passando come parametri "nome stazione", "provincia" e "regione" è stato possibile ottenere lat e lon. Lo script utlizzato è il seguente:

```
string = "station " + str(i[1][1]) + " " + str(i[1][2]) + " " +  str(i[1][3])
location = geolocator.geocode(string)
if location != None:
    print(location.raw)
    staz["Lat"][count] = location.latitude
    staz["Lon"][count] = location.longitude
```

- ***Wikipedia***: in questo caso è stato fatto *web scraping* dal sito, utilizzando il seguente codice, grazie alla libreria **Beautiful Soup**:

```
r = requests.get('https://it.wikipedia.org/wiki/' + nome_stazione)
soup = BeautifulSoup(r.text, "html.parser")
lat = soup.find_all("span", class_="latitude")[0].text
lon = soup.find_all("span", class_="longitude")[0].text
```

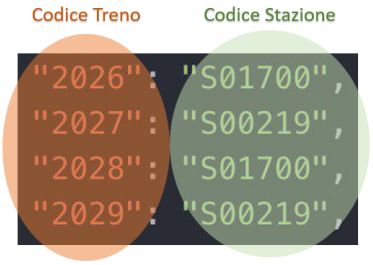
Come ultimo passo, ci siamo resi conto che alcune stazioni erano presenti nel percorso compiuto dai treni, ma non erano stato riportate nel portale ViaggiaTreno e non erano presenti nella nostra base di dati. Quindi, con la stessa procedura elencata sopra, tali campi sono stati opportunamente completati di tutte le informazioni necessarie.

In definitiva, il nostro file csv si presenta con la seguente struttura:

_id	Nome stazione	Regione	Provincia	Latitudine	Longitudine
S05700	ABANO TERME	VENETO	PADOVA	45.355199	11.811533
S01416	ABBADIA LARIANA	LOMBARDIA	LECCO	45.895864	9.335175
S12873	ABBASANTA	SARDEGNA	ORISTANO	40.128801	8.817733
N00018	ABBIATE GUAZZONE	LOMBARDIA	VARESE	45.7037066	8.9204869
S01062	ABBIATEGRASSO	LOMBARDIA	MILANO	45.400631	8.921305

2.3 Creazione file json contenente l'identificativo del treno associato al codice della stazione di partenza

A questo punto, dopo aver completato il file sulle stazioni, l'abbiamo utilizzato per collegare le stazioni ai singoli treni che ogni giorno circolano in Italia. Ci siamo, quindi, creati un file contenente una lista di dizionari (coppie chiave-valore) i quali avessero come chiave i codici identificativi dei treni e come valori l'identificativo della stazione di partenza.
Questo è il risultato ottenuto:



Tale operazione è stata effettuata per ogni giorno del mese di Gennaio, necessaria al fine di considerare tutti i treni (poiché vi è la possibilità che un treno venga effettuato solo in alcuni giorni della settimana).

2.4 Salvataggio dei dati in locale ed in MongoDB

A questo punto, grazie al file di collegamento fra i treni e le stazioni di partenza è stato possibile effettuare una vera e propria fase di *Data Ingestion*. Grazie ad uno script da noi creato, siamo stati in grado di scaricare tramite le API i dati

associati ad ogni treno e a salvarli sotto forma di file json.

```
for i in range(0,len(file)):
    try:
        r = requests.get("http://www.viaggiatreno.it/viaggiatrenonew/resteasy/viaggiatreno/andamentoTreno/"
                         + str(file[treni[i]]) + "/" + str(treni[i]))
```

Tale processo è stato automatizzato grazie ad un timer: alle ore 23:00 di ogni giorno venivano fatte delle richieste al portale in modo da ottenere il maggior numero di informazioni sui treni della giornata. Da notare che dopo le ore 00:00, le informazioni dei treni vengono sovra-scritte da quelle del giorno successivo.
Le informazioni così ottenute, sono state salvate sia in locale che in MongoDB.
Tale procedimento è stato effettuato per ogni giorno del mese di Gennaio 2019, ottenendo complessivamente circa 4 GB di dati.

2.5 Situazione finale

Al termine di questa fase, i dati in nostro possesso sono così composti:

- Una cartella contenente delle sotto cartelle (nominate in base al giorno - es: 01-01-2019) al cui interno vi sono i file json contenenti le informazioni della giornata per ciascun treno;
- Una base di dati su MongoDB contenente tutte i dati dei treni raggruppati in collezioni, una per ogni giorno del mese;
- Un file *stazioni.csv*, completo e senza valori mancanti.

3 Data Preparation

3.1 Struttura iniziale del documento contenente i dati di un treno

Dalle richieste effettuate durante la fase di *Data Ingestion*, abbiamo ottenuto dei documenti in formato json, contenenti ognuno 83 campi: 75 di essi aventi come valore un solo elemento e gli altri 8 aventi come valore una lista di elementi, molti dei quali contenenti valori nulli o non utili al fine del nostro progetto.
Per la creazione della struttura finale del documento, abbiamo preso la decisione di tenere solo alcuni di essi e di ricavarne altri modificando e aggregando quelli già esistenti.
Data la presenza di alcuni casi particolari nelle caratteristiche dei treni (come per esempio la situazione in cui un treno risulta cancellato o parzialmente soppresso), in alcuni file saranno presenti un numero maggiore di campi e, conseguentemente, un maggior numero di informazioni relativi a quel determinato treno.

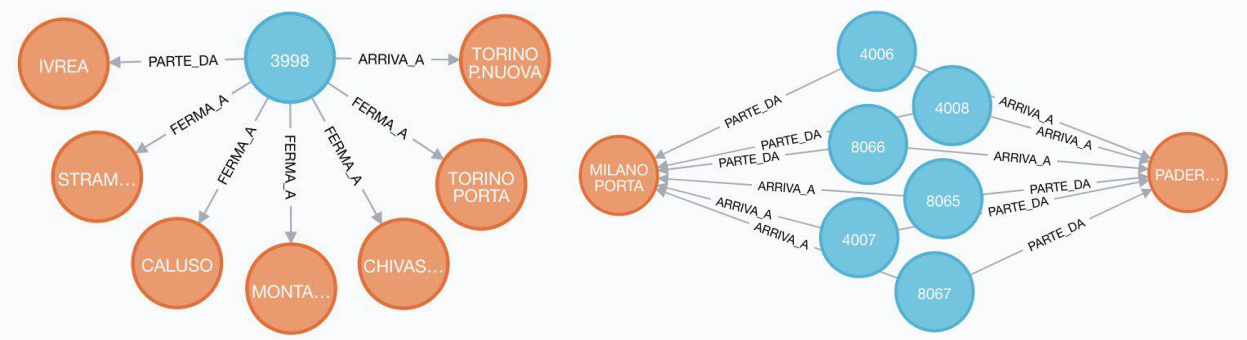
Di seguito, vengono riportati i campi principali. Da notare che gli orari sono espressi in timestamp UNIX con millisecondi e arrotondati ai 30 secondi per difetto. Inoltre, i ritardi sono espressi in minuti interi arrotondati per eccesso.

- **idOrigine / idDestinazione**: codici identificativi delle stazioni di partenza e destinazione;
- **numeroTreno**: codice identificativo del treno;
- **categoria**: categoria del treno (REG, ES*, EN, EC, IC, MET);
- **origineZero / destinazioneZero**: origine teorica e destinazione teorica in caso di treno parzialmente soppresso;
- **origine / destinazione**: nomi delle stazioni di partenza e destinazione;
- **orarioPartenza / orarioArrivo**: orari programmati di partenza dall'origine e arrivo a destinazione, espressi in timestamp;
- **origineEstera / destinazioneEstera / oraPartenzaEstera / oraArrivoEstera**: valorizzati solo per treni internazionali;
- **tipoTreno / provvedimento**: codificano lo stato del treno:
  - **tipoTreno** vale **'PG'** e provvedimento vale **0**: treno regolare;

  - **tipoTreno** vale **'ST'** e provvedimento vale **1**: treno soppresso (in questo caso l'array fermate ha lunghezza 0);
  - **tipoTreno** vale **'PP'** oppure **'SI'** oppure **'SF'** e provvedimento vale **0** oppure **2**: treno parzialmente soppresso (in questo caso uno o più elementi dell'array fermate hanno il campo **actualFermataType** uguale a **3**);
  - **tipoTreno** vale **'DV'** e provvedimento vale **3**: treno deviato con fermate soppresse e con fermate straordinarie;
  - **tipoTreno** vale **'VD'**: treno deviato con fermate soppresse;
- **subTitle** se il treno è parzialmente soppresso (**tipoTreno** in (**'PP'**, **'SI'**, **'SF'**)) contiene una descrizione della tratta cancellata (ad esempio: Treno cancellato da NOVI LIGURE a ALESSANDRIA. Parte da ALESSANDRIA);
- **fermate: array**, un elemento per ogni fermata, con i seguenti campi principali:
  - **id / stazione**: codice e nome della stazione;
  - **tipoFermata**: **'P'** (stazione di origine), **'A'** (stazione di destinazione), **'F'** (fermata intermedia);
  - **ritardoArrivo / ritardoPartenza**: ritardo in minuti di arrivo e partenza alla stazione, in minuti interi;
  - **ritardo**: ritardo in partenza (se **tipoFermata=='P'**) e di arrivo altrimenti, in minuti interi;
  - **arrivoReale / partenzaReale**: orari effettivi di arrivo e partenza nella stazione, in timestamp;
  - **partenza_teorica / arrivo_teorico**: orari teorici di partenza e arrivo nella stazione, in timestamp;
  - **actualFermataType**:
    - **1**: fermata regolare;
    - **2**: fermata non prevista;
    - **3**: fermata soppressa (se **tipoTreno** in (**'PP'**, **'SI'**, **'SF'**));
    - **0**: dato non disponibile (**arrivoReale** e/o **partenzaReale** valgono **null**, può essere perché il treno è ancora in viaggio e deve ancora arrivare nella fermata oppure perché il dato non è stato rilevato);
- **cambiTreno**: lista con eventuali treni straordinari utilizzati per completare il percorso;
- **subtitile**: stringa contenete informazioni aggiuntive su eventuali problematiche.

3.2 Neo4j

Concluse le fasi iniziali di raccolta e preparazione dei dati, abbiamo dovuto scegliere quale tipologia di database NoSQL utilizzare per immagazzinare ed elaborare i nostri dati. La nostra prima scelta, dettata da un'ipotetica rappresentazione dei dati, è caduta su un database a grafo e in particolare su Neo4j. Questa tipologia di database, infatti, è basato sui grafi con etichette (*labeled-property graph model*) in cui abbiamo delle entità, rappresentate da dei nodi, e le relazioni che esistono fra queste entità le quali sono rappresentate da degli archi di collegamento fra i diversi nodi del grafo. La nostra idea, infatti, era quella di costruire l'intera rete ferroviaria italiana: attraverso i nodi rappresentare le stazioni e i singoli treni, mentre le relazioni erano degli archi direzionali che uscivano dai nodi treno ed entravano in tutti i nodi stazioni attraversati da quel treno in una determinata tratta. In questo modo, tutti i nodi treno presentavano solo archi uscenti e i nodi stazione solo archi entranti.
Di seguito vengono riportati due esempi del lavoro che avevamo cominciato ad eseguire:



Tuttavia, durante l'implementazione di tale struttura, ci siamo resi conto di alcune problematiche legate alla tipologia di analisi posta inizialmente. In particolare, non sfruttavamo pienamente le potenzialità della struttura a grafo.

Abbiamo infatti realizzato che una database *document-based* si prestava maggiormente alle nostre analisi, in quanto la struttura iniziale dei nostri dati era molto simile a quella di un documento. Verrà mostrato più avanti come le query, che abbiamo avuto bisogno di realizzare per il nostro studio, siano risultate molto meno complesse di quanto non sarebbero state all'interno di Neo4j. Inoltre, uno degli svantaggi del database a grafo è il fatto che non è possibile effettuare operazioni di UPDATE sui dati, cosa che, invece, ci è stata necessaria durante la fase di manipolazione.

3.3 MongoDB

Avendo deciso di utilizzare un database *document-based* abbiamo optato per MongoDB. I dati raccolti, in formato json, potevano essere facilmente modellati per un sistema che salva i dati in formato BSON (binary json). Inoltre, la creazione di *embedded document* ci ha permesso di salvare le informazioni delle stazioni direttamente nei documenti relativi ai treni, in modo da evitare l'utilizzo di *join table*.
Un altro vantaggio dell'utilizzo di questo database e di questa tipologia di modello è la grande scalabilità che ne consegue. Infatti, il nostro progetto potrebbe essere esteso ai dati di un anno e la struttura da noi utilizzata rimarrebbe la stessa. Un'idea, se si volesse avere una durata annuale dei dati, potrebbe essere quella di crearsi una collezione per ogni mese e dividere la base di dati in *shard*.

Dopo un'attenta analisi dei campi presenti nei dati raccolti, abbiamo definito la struttura del nostro documento. In particolare, in un singolo documento sono contenute tutte le informazioni riguardanti un solo treno e quindi, essendo il numero di treni di cui abbiamo raccolto i dati pari a *236.115*, ci troviamo ad avere lo stesso numero di documenti.

Di seguito, viene mostrato un esempio di struttura del documento che abbiamo deciso di utilizzare. Il treno preso in considerazione è un regionale che parte da *Udine* e arriva a *Venezia Santa Lucia* compiendo *22* fermate prima di arrivare al capolinea.


```
_id: ObjectId("5caccbcc6422510291b3283b")
codice_treno: "REG11077"
categoria: "REG"
ritardo_finale: 0
data: 2019-01-01T00:00:00.000+00:00
origine: "UDINE"
destinazione: "VENEZIA SANTA LUCIA"
tratta: "UDINE-VENEZIA SANTA LUCIA_84"
› fermate: Array
numero_fermate: 22
durata: 135
giorno_settimana: 3
tipo_giorno: "feriale"
stato: "regolare"
```

```
⌄ fermate: Array
  ⌄ 0: Object
      id_stazione: "S03026"
      orario: 2019-01-01T18:31:00.000+00:00
      ritardo: 5
      tipo_fermata: "P"
      nome_stazione: "UDINE"
      provincia: "UDINE"
      regione: "FRIULI VENEZIA GIULIA"
      lat: "46.055791"
      lon: "13.242003"
  ⌄ 1: Object
      id_stazione: "S02832"
      orario: 2019-01-01T18:39:00.000+00:00
      ritardo: 6
      tipo_fermata: "F"
      nome_stazione: "BASILIANO"
      provincia: "UDINE"
      regione: "FRIULI VENEZIA GIULIA"
      lat: "46.012737"
      lon: "13.109233"
  ⌄ 21: Object
      id_stazione: "S02593"
      orario: 2019-01-01T20:46:00.000+00:00
      ritardo: 0
      tipo_fermata: "A"
      nome_stazione: "VENEZIA SANTA LUCIA"
      provincia: "VENEZIA"
      regione: "VENETO"
      lat: "45.441569"
      lon: "12.320882"
```


Facciamo adesso un'analisi di tutti i diversi campi contenuti nella struttura del documento e teniamo l'esempio visto sopra come linea guida.
Ricordiamo che la struttura iniziale dei nostri dati presentava 83 campi diversi di cui molti presentavano valori nulli o erano inutilizzabili. Per questo abbiamo deciso di tenerne solo alcuni e di manipolarne altri, unendoli fra loro, per crearne di nuovi con informazioni aggiuntive che potessero tornarci utili al fine del nostro studio.

Come prima cosa troviamo "**_id**" che è la chiave primaria di ogni documento e viene generata automaticamente da MongoDB nel momento dell'inserimento all'interno del database.

Il campo "**codice_treno**" è stato creato da noi come combinazione fra la categoria (che viene riportata anche subito dopo) e il numero del treno.

Il "**ritardo_finale**" è uno dei dati fondamentali per la nostra analisi perchè rappresenta il ritardo del treno a destinazione.

La "**data**" è il risultato della conversione di quella presente nella struttura iniziale che era espressa in *timestamp UNIX* con millisecondi e arrotondata ai 30 secondi per difetto. Nel nostro caso l'abbiamo trasformata in un formato *datetime* in cui vengono espresse sia la data che l'ora. Grazie a questa conversione siamo riusciti anche ad ottenere il giorno della settimana (campo "**giorno_settimana**") attraverso il comando "**$weekdays**" di MongoDB. Da notare che questo dato viene indicato attraverso gli interi di una sequenza numerica che va da 1 a 7, dove 1 corrisponde a domenica e 7 a sabato (MongoDB assume convenzionalmente questo ordine della settimana).

Sempre procedendo in questa direzione, siamo riusciti a risalire e, quindi ad aggiungere, anche il "**tipo_giorno**" che assume i valori **"festivo"** o **"feriale"**. L'unico caso particolare è l'1 Gennaio che risulta essere un giorno festivo, nonostante sia un martedì.

Vengono poi riportati l'"**origine**" e la "**destinazione**". Questi due campi ci sono stati fondamentali per la creazione del nuovo campo "**tratta**". Per fare questo, abbiamo fatto una scansione di tutti i nostri dati raccolti, in modo tale da individuare una maniera per raggruppare determinati treni sotto uno stesso nome. In particolare, abbiamo considerato due treni come corrispondenti alla stessa tratta se possedevano le seguenti caratteristiche:

1. Stessa partenza
2. Stesso arrivo
3. Stessa categoria
4. Stessa sequenza di fermate intermedie

Questo nuovo campo ci è stato utile soprattutto nella parte di visualizzazione che abbiamo eseguito in seguito.
Per la descrizione del percorso effettuato dal treno preso in considerazione, abbiamo sfruttato la possibilità di innestare i documenti. Abbiamo introdotto il campo "**numero_fermate**", che rappresenta il numero totale di fermate compiute dal treno, e "**fermate**". Quest'ultimo è un array di lunghezza pari a **numero_fermate** che contiene tutte le informazioni relative alle stazioni in cui transita il treno.
Queste informazioni sono state ricavate dal csv contenente i dati sulle stazioni ferroviarie italiane creato precedentemente attraverso il codice identificativo della stazione. In questa maniera, abbiamo riportato per ogni singola fermata il **nome_stazione**, la **provincia**, la **regione** e **latitudine** e **longitudine** corrispondenti.
Inoltre, sempre all'interno di ogni fermata abbiamo riportato l'**orario**, in *datetime* convertito nello stesso modo della data, e abbiamo aggiunto il **tipo_fermata** che assume valori **"P"**, **"F"** e **"A"** per distinguere le fermate intermedie dalla partenza e dall'arrivo.
Infine, il campo più importante che abbiamo inserito è quello riguardante il **ritardo** di ogni singola fermata. Per riportare questo campo abbiamo dovuto fare alcune assunzioni per fare in modo di non avere errori. Per tutte le fermate che avevano tipo fermata uguale a **"P"** abbiamo utilizzato il ritardo in partenza (**ritardoPartenza**) presente nei json iniziali. Per quanto riguarda le fermate intermedie invece abbiamo dovuto fare un'analisi più dettagliata e prendere delle decisioni. Infatti, dato che ad ogni fermata era presente sia il ritardo in arrivo che il ritardo in partenza abbiamo deciso di considerare come ritardo della fermata il ritardo in partenza. Questo significa che se un treno è rimasto fermo in una stazione per più tempo di quello previsto il ritardo viene accumulato su quella singola fermata. Gli arrivi, invece, sono gli unici in cui viene considerato come ritardo il ritardo in arrivo (**ritardoArrivo**) in quanto, per ovvi motivi, il ritardo in partenza non è presente.

Tornando alla struttura del documento iniziale, rimangono solamente due campi da descrivere.
Il primo è il campo "**durata**", il quale rappresenta in minuti il tempo necessario per percorrere il percorso dalla stazione di origine a quella di destinazione. Per calcolare questo campo abbiamo effettuato la differenza fra l'orario in arrivo nella stazione di destinazione e l'orario in partenza della stazione di origine (entrambi dati che si trovano all'interno dell'array fermate, come abbiamo già detto).

Infine, l'ultimo campo rimanente è lo "**stato**". Questo ci permette di distinguere fra i treni che hanno svolto il loro percorso regolarmente e quelli che hanno avuto delle sospensioni o delle cancellazioni all'interno della tratta. Infatti, questo campo può assumere i seguenti valori: **"regolare"**, **"parzialmente soppresso"** e **"cancellato"**. Questi ultimi due valori sono stati ottenuti manipolando i campi "**tipoTreno**" e "**provvedimento**" presenti nei dati iniziali e presentati precedentemente.
Per quanto riguarda i treni parzialmente soppressi abbiamo aggiunto i campi "**destinazioneTeorica**" e "**origineTeorica**". Questo è necessario in quanto, essendo il treno soppresso solo in parte, esso avrà un'origine o una destinazione differenti da quelle riportate in questi campi.
La grande differenza nei treni cancellati, invece, è il fatto che il campo "fermate", che prima era un array, qui è identificato dalla stringa **"Non rilevate"**, in quanto, ovviamente, il treno non ha potuto effettuare nessuna fermata.
Tuttavia, esistono dei treni che, nonostante abbiano come stato **"regolare"**, presentano la stringa **"Non rilevate"** nelle fermate, a causa di una mancata rilevazione dei dati.
Oltre all'indice generato automaticamente da mongoDB sul campo _id abbiamo deciso di inserire altri tre indici: su durata, tratta e categoria per migliorare l'efficienza delle query necessarie per ottenere le informazioni da visualizzare.

3.3.1 Gestione delle anomalie

Come avevamo già accennato inizialmente, abbiamo dovuto prestare particolare attenzione ai dettagli in quanto i dati che abbiamo raccolto presentavano, in alcuni casi, qualche anomalia. Dopo le dovute ricerche, abbiamo supposto che la maggior parte di queste anomalie fossero dovute a dei malfunzionamenti nelle rilevazioni. In alcuni casi, all'interno del percorso di un treno c'era una singola fermata che presentava un ritardo eccessivamente elevato rispetto alla fermata precedente e a quella successiva oppure aveva un orario che risultava improbabile. Data la scarsità di queste situazioni, circa una ventina fra più di *200 mila treni*, abbiamo preso la decisione di non considerare questi valori anomali nella nostra analisi e li abbiamo quindi eliminati.

3.4 Query MongoDB e osservazioni

In questa nuova sezione, mostriamo alcuni esempi delle *query* che abbiamo dovuto effetuare durante il nostro percorso. Grazie alla struttura basata sui documenti, è stato relativamente semplice ricavare le informazioni di cui avevamo bisogno per la nostra analisi, cosa che sarebbe risultata molto più difficile in una struttura a grafo. Quest'ultima infatti, nonostante permettesse di aggiungere degli attributi sia ai nodi che agli archi, non ci permetteva di accedere alle informazioni necessarie allo stesso modo e con la stessa facilità di MongoDB.

Per effettuare le *query*, abbiamo utilizzato la libreria di *PyMongo* all'interno di *Python* che ci permetteva di interrogare il database in *MongoDB* senza dover utilizzare direttamente il terminale.
Nel primo esempio, stiamo interrogando la nostra base di dati per ottenere i cinque treni appartenenti alla categoria dell'alta velocità ("ES*") che presentano il ritardo medio maggiore.

```
pipeline = [
        {"$match": {"categoria": "ES*"}},
        {"$group": {"_id": "$codice_treno",
                    "ritardo_medio": {"$avg": "$ritardo_finale"}}},
        {"$sort": SON([("ritardo_medio", -1)])}
]
pd.DataFrame(list(db.trenitalia.aggregate(pipeline))[:5])
```

Il risultato viene mostrato nella seguente tabella in cui abbiamo il codice identificativo del treno e anche il ritardo medio corrispondente. Possiamo osservare che il treno peggiore in assoluto presenta un ritardo medio di ben 97 minuti.

	_id	ritardo_medio
0	ES*8317	97.800000
1	ES*8315	29.935484
2	ES*8325	29.925926
3	ES*9552	29.263158
4	ES*9560	29.193548

Nella seconda *query* abbiamo deciso di sfruttare la possibilità di raggruppare i treni e l'abbiamo fatto attraverso il comando *group_by*. In particolare, abbiamo raggruppato su due livelli differenti. In primo luogo, abbiamo suddiviso i dati per categoria di treno e a loro volta li abbiamo suddivisi secondo i diversi valori che possono essere assunti dal campo "stato", che ricordiamo essere: "regolare", "parzialmente soppresso" e "cancellato".

```
pipeline = [
        {"$group": {"_id": {"categoria":"$categoria", "stato":"$stato"},
                    "totale": {"$sum": 1}}},
        {"$sort": SON([("_id.categoria", -1)])}
]
lista = (list(db.trenitalia.aggregate(pipeline)))
```

Nel risultato finale vengono mostrati i due campi nominati sopra e il numero totale di treni per ogni coppia categoria-stato.

	categoria	stato	totale
0	REG	regolare	211759
1	REG	cancellato	2167
2	REG	parzialmente soppresso	1807
3	MET	parzialmente soppresso	143
4	MET	cancellato	135
5	MET	regolare	6344
6	IC	cancellato	13
7	IC	regolare	3464
8	IC	parzialmente soppresso	30

	categoria	stato	totale
9	ES*	cancellato	4
10	ES*	parzialmente soppresso	22
11	ES*	regolare	8557
12	EN	cancellato	14
13	EN	regolare	185
14	EN	parzialmente soppresso	11
15	EC	cancellato	4
16	EC	regolare	1451
17	EC	parzialmente soppresso	5

In quest'ultima *query* mostriamo l'utilizzo del comando "$unwind", che ci è risultato fondamentale durante le nostre analisi. Questo comando, infatti, ci permette di "scomporre" l'array fermate in più documenti e di indagare anche i campi presenti all'interno di questi sotto-documenti. In questo modo ci è stato possibile risalire alle informazioni riguardanti i ritardi delle singole stazioni.
Nell'esempio riportato stiamo chiedendo di ricercare i treni che attraversano almeno una volta una stazione della Lombardia. Dopo aver raggruppato per il nome della stazione calcoliamo il ritardo medio che viene accumulato in ogni singola stazione.

```
pipeline = [
        {"$unwind": "$fermate"},
        {"$match": {"fermate.regione":"LOMBARDIA"}},
        {"$group": {"_id": "$fermate.nome_stazione",
                    "ritardo_medio": {"$avg": "$fermate.ritardo"}}},
        {"$sort": SON([("ritardo_medio", -1)])}
]
pd.DataFrame(list(db.trenitalia.aggregate(pipeline))[:5])
```

Anche in questo caso, come nella prima *query*, mostriamo le cinque stazioni in Lombardia che presentano un ritardo medio maggiore. Questo ci è possibile grazie al fatto che abbiamo ordinato le stazioni in ordine decrescente di ritardo. Dalla tabella riportata possiamo osservare che le peggiori stazioni lombarde hanno un ritardo medio compreso fra i 10 e i 16 minuti.

	_id	ritardo_medio
0	MANERBIO	15.344262
1	VEROLANUOVA	15.163934
2	S.STEFANO LODIGIANO	14.031250
3	CHIARI	11.619835
4	CHIUDUNO	10.475806

Parte II
Data Visualization

Una volta definita l'architettura della nostra base di dati, siamo passati alla parte di *Data Visualization*. Abbiamo elaborato quattro visualizzazioni per descrivere i risultati più interessanti del nostro studio.

4 Strumenti utilizzati

Per eseguire questo lavoro ci siamo serviti di più strumenti diversi. Gli strumenti utilizzati sono i seguenti:

- Tableau Desktop
- JavaScript Charts Maps - amCharts
- Seaborn e Matplotlib (librerie di python)
- ggplot2 (libreria di R)

Data la grande quantità di dati, abbiamo utilizzato *MongoDB Connector* per rendere disponibile l'intera base di dati su Tableau. Spesso, però, tale metodologia presentava delle problematiche. Infatti, nel caso in cui venivano richiesti dei dati aggregati su livelli differenti, essa risultava essere molto lenta e, per questo motivo, ci siamo creati dei dataset *ad hoc*, interrogando la base di dati con PyMongo, per realizzare le nostre infografiche.

5 Le infografiche

Le infografiche elaborate sull'andamento del traffico ferroviario nazionale consentono di approfondire ed evidenziare un aspetto cruciale dell'infrastruttura urbana del trasporto, ossia l'insufficiente costanza nella puntualità dei treni sulla rete nazionale e l'inevitabile ritardo accumulato dagli stessi.
Abbiamo iniziato focalizzando la nostra attenzione sul traffico ferroviario per poi effettuare una panoramica generale sui ritardi nelle varie provincie italiane. Infine, abbiamo concluso con due infografiche che ci hanno permesso di entrare maggiormente nel dettaglio facendo una divisione per le due categorie più importanti (regionali ed alta velocità).

Nelle prossime pagine verranno descritte nei particolari le diverse infografiche e, infine, verranno riportati i risultati e le statistiche sui questionari e sui test effettuati per verificare la qualità del lavoro eseguito.

5.1 Traffico ferroviario del mese di Gennaio 2019

Nel mese di Gennaio 2019 sono stati monitorati i dati relativi al servizio della Rete Ferroviaria Italiana, includendo sia quelli delle reti ad essa interconnesse che quelli delle reti isolate. Di seguito, possiamo vedere graficamente il numero totale di treni circolati in questo periodo, considerando i treni in partenza ed in arrivo nella regione, includendo tutte le categorie (Alta Velocità, Regionali, Eurocity, Intercity, Euronight e Met).

Descrizione dell'infografica

Lungo la circonferenza del *chord diagram* sono presenti le regioni italiane, ordinate in senso orario in base al numero totale dei treni in partenza ed in arrivo.
Gli archi che collegano le regioni (e "ESTERO") rappresentano il numero di treni che partono da una regione ed arrivano in un'altra (o nella stessa).
La scala dei colori è ordinata in base al numero di treni totali (in partenza e in arrivo) della regione, dove il colore più scuro rappresenta la regione col maggior numero di treni totali, diminuendo l'intensità del colore fino alla regione col numero minore.

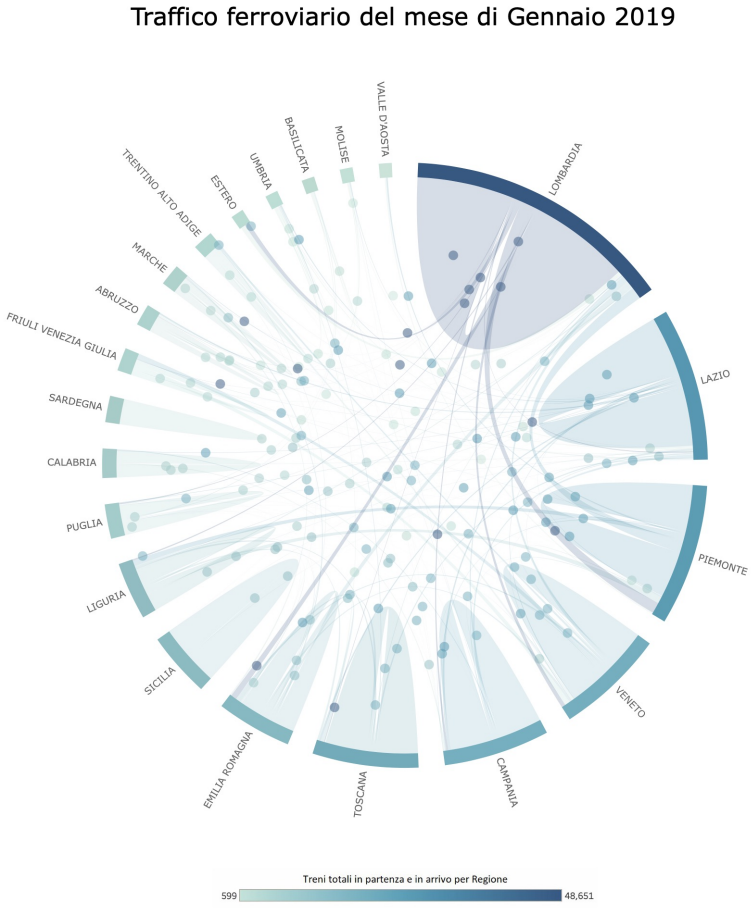


Figura 1: Clicca sull'immagine o sul seguente link: `https://aleriboni.github.io/Data-Viz/infografica1.html` per visualizzare l'infografica interattiva

È possibile notare che le regioni col maggior traffico ferroviario sono – in ordine di numerosità - la Lombardia, il Lazio, il Piemonte, il Veneto e le Campania. La somma dei treni circolati in queste regioni corrisponde a circa la metà del traffico ferroviario italiano.

5.2 Panoramica generale sui ritardi

Analizzato il traffico ferroviario, è possibile fornire una panoramica generale sui ritardi mediani per provincia, includendo tutte le categorie sopracitate.

Descrizione dell'infografica

Lungo la circonferenza più esterna del *radar timeline chart* sono presenti le provincie italiane, raggruppate per regione. Queste ultime sono in ordine alfabetico.
I *barchart* presenti nella circonferenza più interna rappresentano il ritardo mediano per giorno e per provincia.
Il range creato dalle linee ondulate rappresenta l'asse troncato.
Per cambiare giorno è necessario utilizzare la barra in alto, facendo scorrere il cursore a destra o a sinistra.
Inoltre, è possibile entrare nel dettaglio di una determinata regione, cliccando su di essa. In questo modo, vengono visualizzate tutte le province presenti in quella regione e i ritardi relativi.

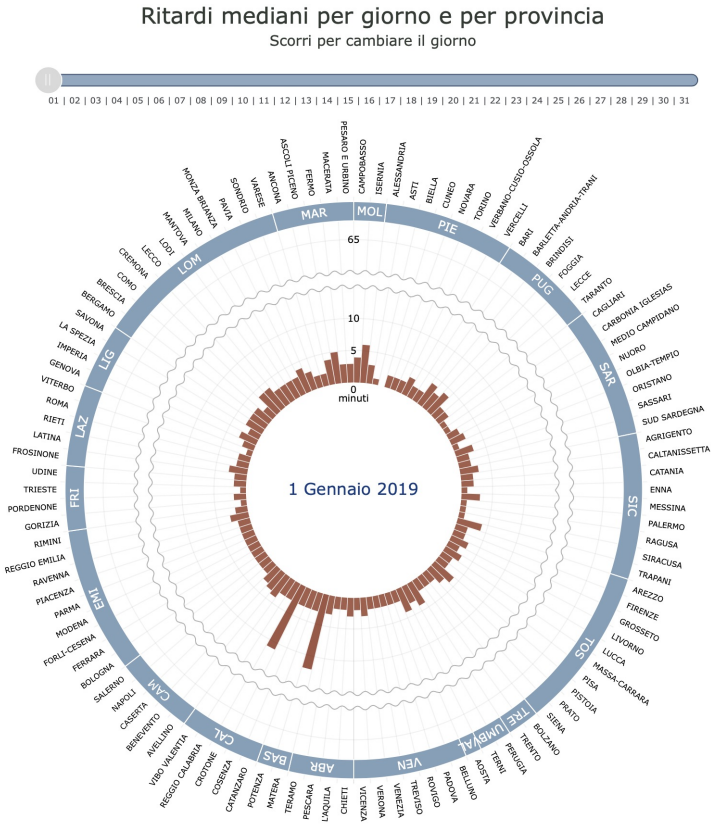


Figura 2: Clicca sull'immagine o sul seguente link: `https://aleriboni.github.io/Data-Viz/infografica2.html` per visualizzare l'infografica interattiva

Da questa infografica è possibile vedere che nella provincia di Isernia, situata in Molise, il ritardo mediano nella giornata del 3 gennaio ha raggiunto 68 minuti. È inoltre possibile verificare ritardi evidenti in Puglia, Sicilia e Basilicata. Situazione simile per la giornata del 4 gennaio: in provincia di Enna (Sicilia) si verifica il ritardo mediano peggiore con circa 55 minuti di ritardo. A seguire alcune province del Molise e della Sardegna. Nel resto del mese la situazione si stabilizza: spesso i ritardi non superano di molto i 5 minuti.

5.3 Focus sui treni regionali

Coi dati raccolti è stato possibile compiere delle analisi sui treni Regionali, i quali ricoprono una percentuale importante della rete ferroviaria Italiana.

Descrizione dell'infografica

In questa infografica sono rappresentati i ritardi medi suddivisi per giorno e per regione. In alto a destra è possibile selezionare il giorno della settimana. Dalla cartina,invece, è possibile selezionare la regione, dove il colore rappresenta il ritardo medio del giorno.
La scala dei colori è ordinata in base al ritardo medio, dove il colore più scuro rappresenta la regione col ritardo medio maggiore, diminuendo l'intensità del colore fino alla regione con ritardo medio minore.
Infine, è possibile osservare i dati relativi al numero medio dei treni, alla percentuale di treni con ritardo medio maggiore di zero e quella con ritardo medio maggiore di cinque. Anche questi dati variano in base alla regione selezionata.

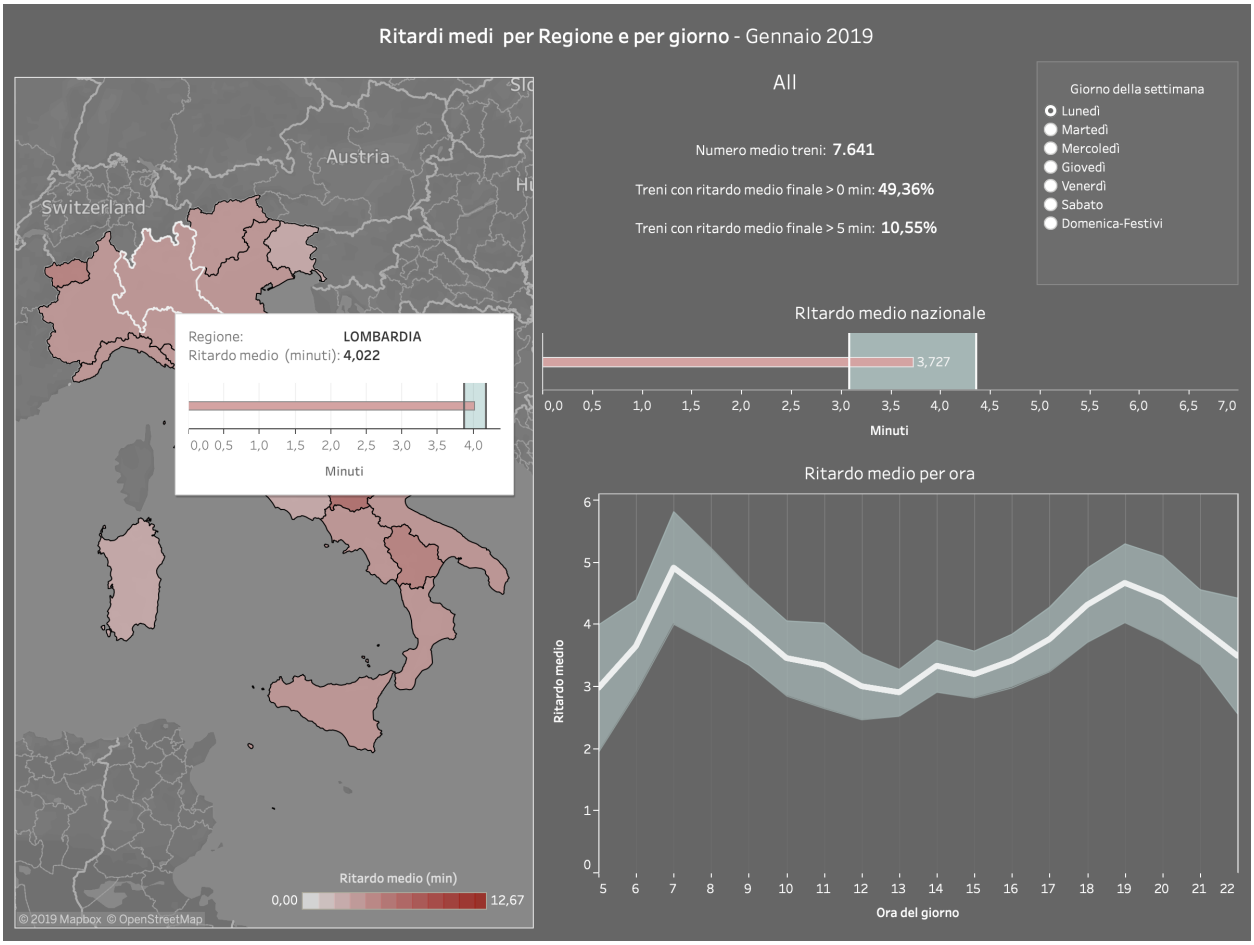


Figura 3: Clicca sull'immagine o sul seguente link: `https://aleriboni.github.io/Data-Viz/infografica3.html` per visualizzare l'infografica interattiva

È possibile osservare che, nei giorni festivi, la situazione dei regionali è mediamente la migliore del mese. Questo è anche dovuto al fatto che i treni in circolazione erano circa la metà di quelli nei giorni feriali.
Il giorno mediamente peggiore, invece, risulta essere il venerdì, in cui i treni regionali con ritardo medio maggiore di 5 minuti è circa del 13%, mentre considerando i ritardi medi maggiori di zero, la percentuale supera il 50%.

5.4 Focus sui treni ad alta velocità

La questione relativa ai treni ad alta velocità merita un capitolo a parte. Essi, infatti, permettono di collegare molte città italiane riducendo drasticamente il tempo di percorrenza normalmente impiegato dai treni regionali. Il grafico che ne deriva mette in evidenza il profondo divario di minuti di ritardo mediani a seconda della tratta.

Descrizione dell'infografica

Come prima cosa, è possibile selezionare l'origine e/o la destinazione, in modo da filtrare le tratte dei treni ad alta velocità.
Sulla cartina è possibile visualizzare i percorsi. Il *bubble chart* permette di visualizzare il numero di treni che effettuano tale percorso. La scala di colori utilizzata per entrambi varia dal giallo (in anticipo) al blu (in ritardo), in base al ritardo mediano finale.
Selezionando uno specifico percorso (o sulla cartina o sul *bubble chart*) è possibile visualizzare il ritardo mediano in ogni singola stazione.
Infine, è possibile osservare i dati relativi alla durata, al ritardo mediano finale e alla percentuale di treni con ritardo mediano finale maggiore di 5 minuti, in relazione alle tratte selezionate.

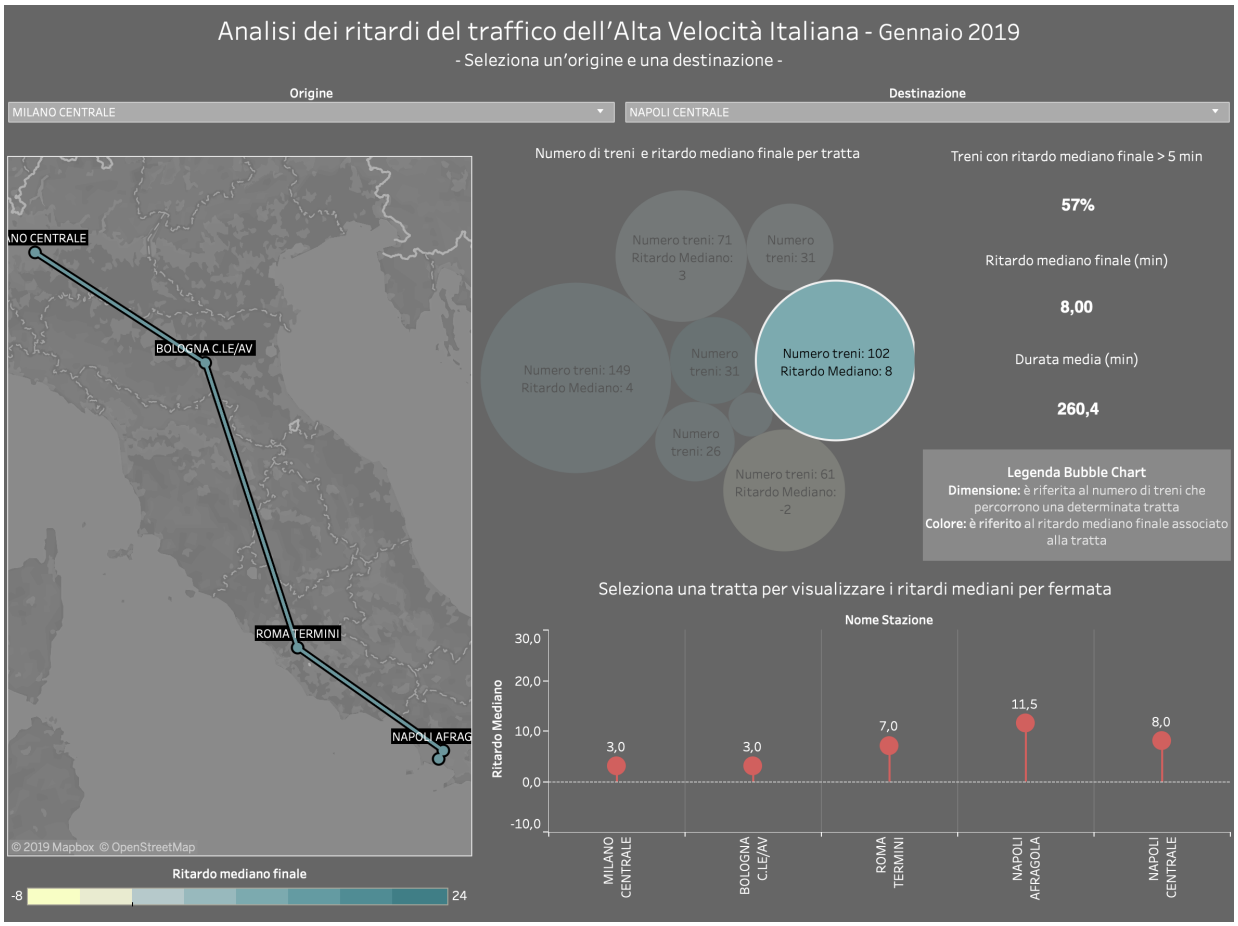


Figura 4: Clicca sull'immagine o sul seguente link: `https://aleriboni.github.io/Data-Viz/infografica4.html` per visualizzare l'infografica interattiva

Si può notare che nel mese di gennaio i treni in partenza da Roma e che arrivano a Bari sono i peggiori: il ritardo mediano è di 24 minuti e il 93% dei treni che la percorrono supera la soglia dei 5 minuti di ritardo convenzionali.
Diverse tratte, invece, arrivano in anticipo rispetto all'orario programmato. Questi anticipi, rappresentati come

ritardi negativi, potrebbero essere confortanti se non si andassero ad osservare i ritardi ottenuti per ciascuna stazione: se si prende in considerazione, ad esempio, una delle tratte della Milano-Lecce, è possibile osservare che i treni arrivano effettivamente in anticipo di quasi 5 minuti a destinazione, ma se si osservano singolarmente le stazioni precedenti all'arrivo si può notare che sono stati registrati ritardi mediani considerevolmente maggiori di zero. Dunque, l'informazione del ritardo mediano a destinazione potrebbe non rappresentare appieno la situazione reale delle tratte e sorge peraltro il dubbio che queste tratte siano state tarate in modo "ambiguo".

6 Valutazione della qualità delle infografiche

6.1 Problematiche

Durante la progettazione delle infografiche un contributo importante è stato dato degli utenti: attraverso le loro osservazioni in merito ad alcuni dettagli delle infografiche ci è stato possibile migliorarle ed aumentarne l'efficacia e l'interpretabilità.
Per semplicità, verranno chiamate "Infografica 1", "Infografica 2", "Infografica 3" e "Infografica 4" in ordine di come sono state presentate precedentemente.

In particolare, sono stati riscontrati dei problemi, alcuni dei quali risolti:

- **Infografica 1**
  1. l'ordine delle regioni lungo la circonferenza era casuale ed ordinandole in base al numero totale per regione è ora più comprensibile confrontare il numero di treni delle diverse regioni;
  2. a volte risulta difficile individuare il numero di treni tra una regione ed un'altra: la fitta rete di archi non ne consente la completa comprensione;
- **Infografica 2**
  1. la barra per scorrere i giorni era inizialmente priva di scala contente i giorni del mese: una volta introdotta è risultata essere più chiara;
  2. l'asse troncato rende a volte difficile l'usabilità dell'infografica.
- **Infografica 3**
  1. in alcuni giorni della settimana è difficile osservare le differenze sul coropleto ed è necessario leggere il tooltip per individuare la regione peggiore;
  2. inizialmente il *line chart* era posizionato sotto la cartina e l'utente spesso non lo notava: modificandone la posizione è risultato più chiaro;
  3. le informazioni del numero medio di treni e le percentuali di treni con ritardo medio superiore a zero e a cinque minuti, per alcuni utenti, risultano essere poco chiare: in particolare, non è chiaro che se si seleziona una regione sul coropleto, i dati descrivono la regione selezionata. Viceversa, se non viene selezionata alcuna regione, i dati si riferiscono alle medie nazionali.
- **Infografica 4**
  1. l'utente spesso tende ad utilizzare la cartina per selezionare le tratte anziché utilizzare gli appositi spazi dedicati alla selezione dell'origine e destinazione;
  2. il *bubble chart* non viene quasi mai utilizzato poiché non se ne capisce l'utilità. In particolare, non è sempre chiaro che vi possono essere più tratte con la stessa origine e destinazione;
  3. la legenda del *bubble chart* non viene quasi mai notata;
  4. non è chiaro che la tratta è più facilmente selezionabile sul *bubble chart*, anziché sulla cartina.

6.2 Test dell'utente

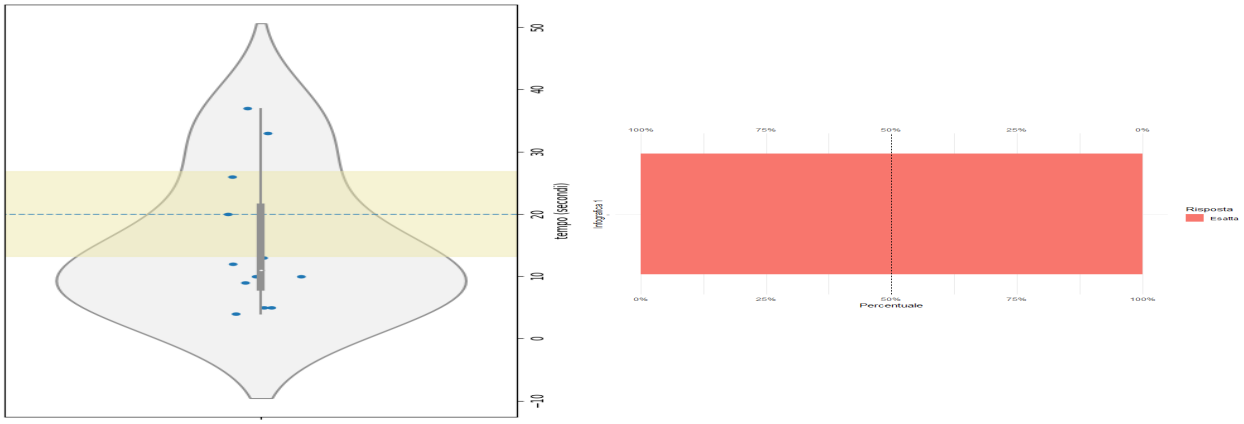
Abbiamo sottoposto gli utenti a delle domande per testare l'effettiva comprensibilità ed usabilità delle infografiche. Abbiamo inoltre calcolato il tempo di cui ogni utente ha avuto bisogno per svolgere il task. In seguito, abbiamo esposto i risultati ottenuti attraverso dei *violin diagram* e degli *stacked bar chart*. I primi sono relativi al tempo di esecuzione, mentre i secondi si riferiscono all'*error rate*, cioè la frequenza delle risposte sbagliate sulle risposte totali. Nei *violin diagram* è stato inserito un tempo di risposta ottimale definito a priori.
Dal tempo ottimale ci siamo creati un range secondo la distribuzione normale. In particolare gli estremi del range sono stati calcolati aggiungendo e sottraendo la deviazione standard dal valore ottimale.

Nella scelta dei task, abbiamo preso la decisione di rendere le domande man mano più articolate al susseguirsi delle infografiche, seguendo un modello top-down. Questo modello è stato l'idea base di tutto il nostro progetto in quanto è basato sul concetto di partire da una formulazione generale del problema e scendere successivamente nel dettaglio.

Di seguito sono riportate le domande poste ad un campione di 12 persone ed i relativi risultati.

Infografica 1

- *Qual è la regione col maggior numero di treni in arrivo ed in partenza? E qual è quella col minor numero?*

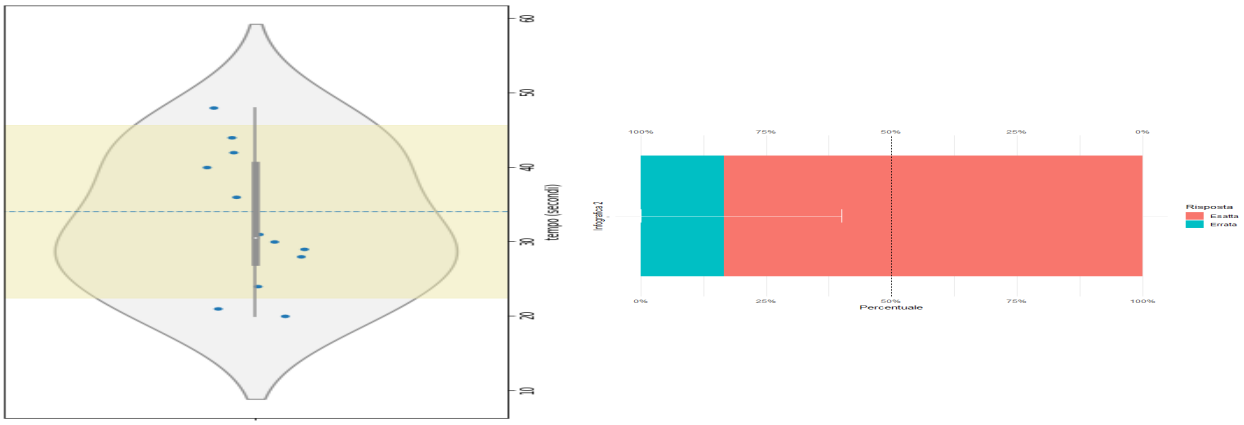


Dai risultati del *violin diagram*, possiamo notare come la maggior parte degli utenti non rientri all'interno dell'intervallo di tempo prefissato. Questo è stato causato dal fatto che, essendo questa la prima infografica presentata, alcuni utenti hanno preso il task come una sfida e hanno cercato velocemente la risposta senza prestare particolare attenzione al resto. Al contrario, ci sono stati un paio di utenti che hanno preferito dare uno sguardo più accurato alla visualizzazione prima di rispondere.

I risultati esposti dallo *horizontal 100% stacked bar chart* evidenziano il fatto che nessuno degli utenti presi in esame abbia dato una risposta errata, cosa che ci aspettavamo data la facilità della domanda.

Infografica 2

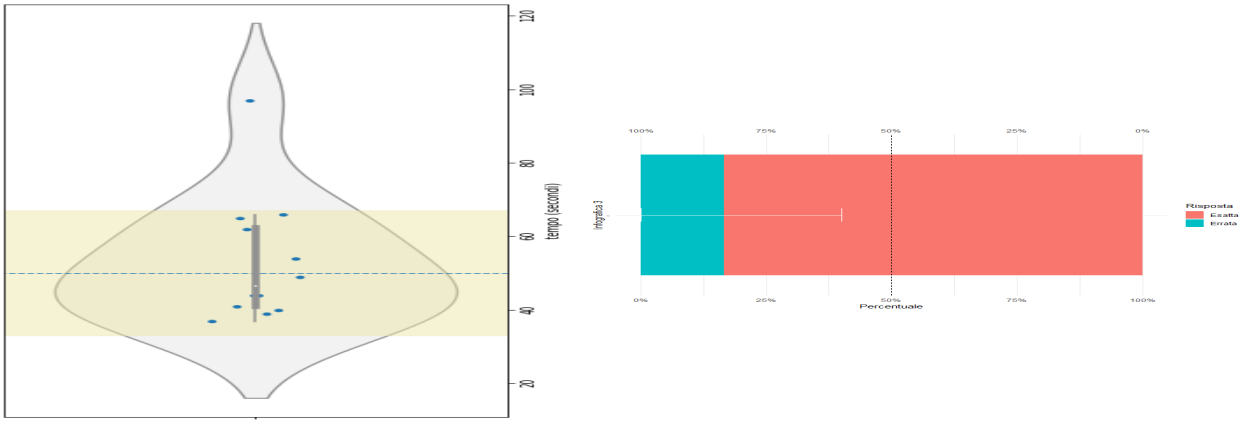
- *Qual è la provincia che presenta il ritardo mediano maggiore il 12 gennaio?*



In questo secondo task, nel *violin diagram* possiamo notare come molti più utenti rientrino all'interno del range di tempo calcolato. Sui 12 a cui è stata sottoposta la domanda, solamente 3 ci hanno messo più o meno tempo di quello previsto.
Questo è dovuto al fatto che, per trovare la risposta, l'utente aveva bisogno di interagire con questa visualizzazione, in quanto doveva spostare il cursore posto sulla barra temporale per trovare il giorno di gennaio richiesto. Al contrario, nella precedente bastava osservare l'infografica per risolvere il task.
Sempre diversamente dalla prima, in questo caso ci sono state delle risposte errate. Tuttavia, l'error rate non è abbastanza elevato da risultare significativo, in quanto l'intervallo di confidenza non supera il livello del 50%.

Infografica 3

- *Qual è la regione che presenta un ritardo medio maggiore verificatosi durante i mercoledì del mese di gennaio? Qual è la fascia oraria in cui si verifica il ritardo maggiore in questa regione?*

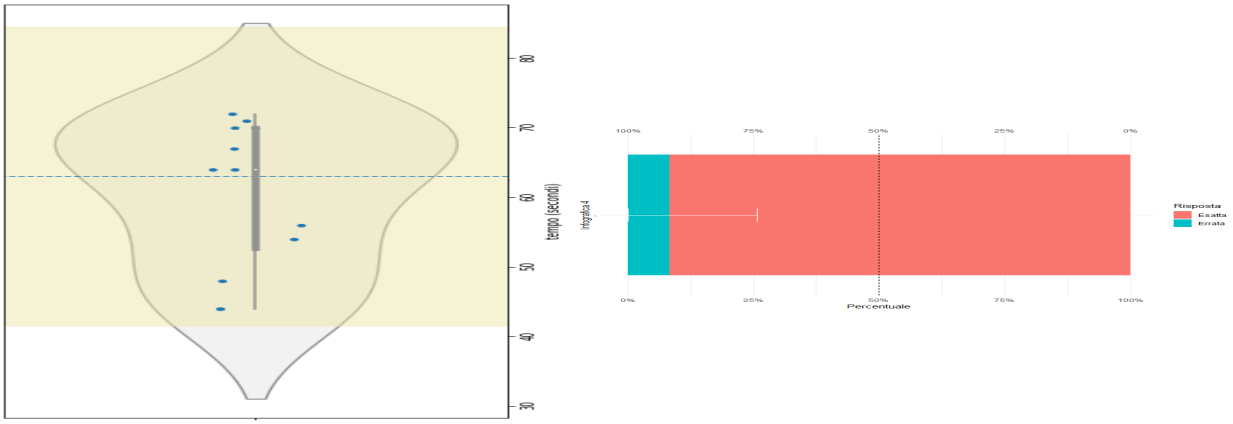


Dal *violin diagram*, possiamo notare che ancora più persone rientrano all'interno del range temporale. Infatti, solamente un utente ci ha messo più tempo del previsto per rispondere alla domanda. Questo utente è anche uno degli utenti che ha dato una risposta errata.

Tuttavia, nonostante la presenza di due risposte errate, anche in questo caso, l'error rate non è abbastanza elevato da risultare significativo, in quanto l'intervallo di confidenza non supera il livello del 50%.

Infografica 4

- *Considera il tragitto che ha come origine "Milano Centrale" e come destinazione "Napoli Centrale". Dopo aver selezionato il percorso che compie tale tragitto col ritardo mediano maggiore, qual è la stazione, all'interno di tale percorso, in cui si verifica il ritardo mediano maggiore?*



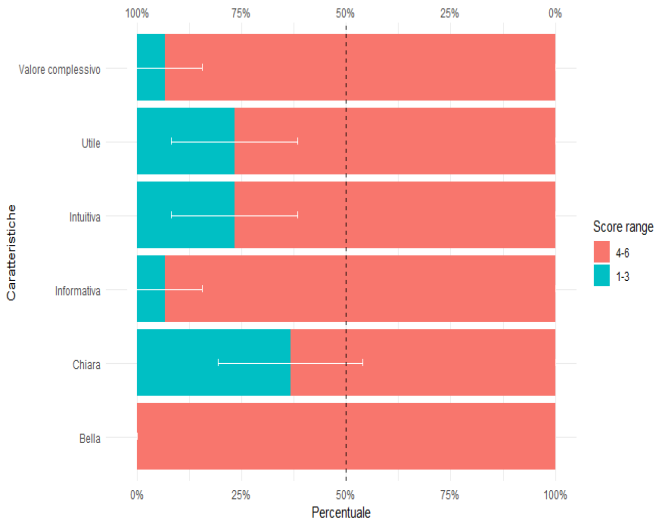
Quest'ultima visualizzazione presentava una domanda più articolata e che necessitava una comprensione maggiore del suo funzionamento. Nonostante questo, tutto gli utenti sono rimasti nell'intervallo di tempo previsto.

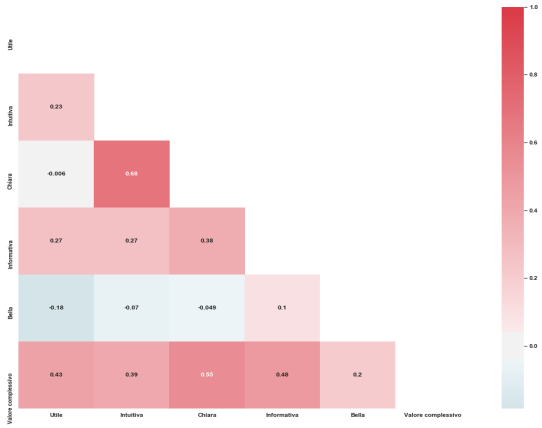
Al contrario delle due infografiche precedenti, in questa solamente una persona ha dato una risposta sbagliata.
Inoltre, come in tutti gli altri casi, la presenza di errori nelle risposte non porta ad avere un error rate significativo.

6.3 Valutazione questionari psicometrici

Oltre ad aver effettuato gli "*users test*", abbiamo selezionato un campione di 30 persone, differenti dalle 12 già prese in esame, e abbiamo sottoposto loro il questionario psicometrico "*Cabitza-Locoro*" per valutare la qualità del nostro lavoro.

Infografica 1

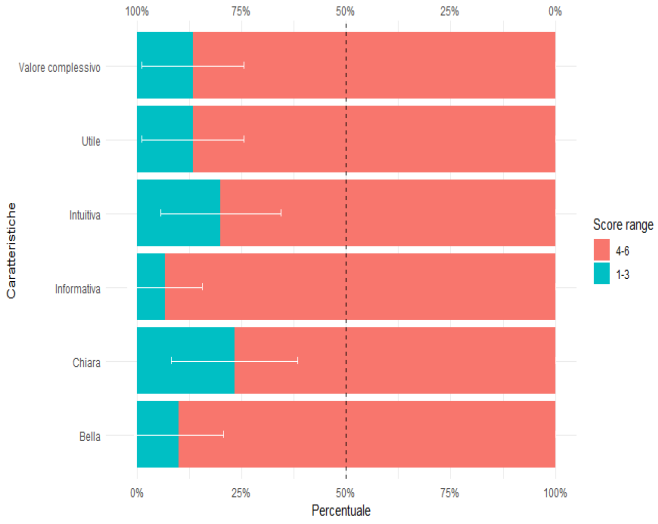


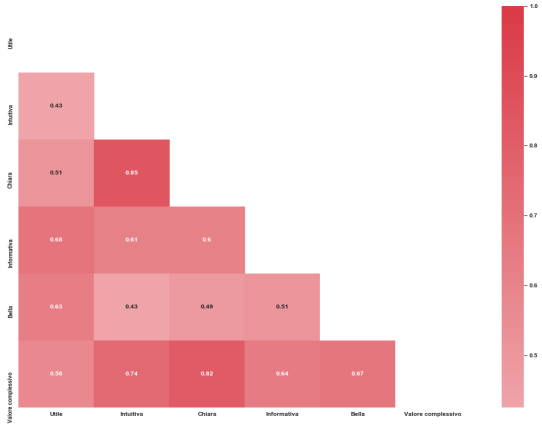


I risultati per questa infografica sono soddisfacenti: solo il campo "Chiara" ha un intervallo di confidenza che interseca la linea del 50%, ossia non è completamente evidente che questa infografica sia effettivamente "Chiara". Negli altri campi la prevalenza è netta.

Non vengono riscontrare particolari correlazioni: la correlazione più alta risulta essere tra "Chiara" ed "Intuitiva".

Infografica 2

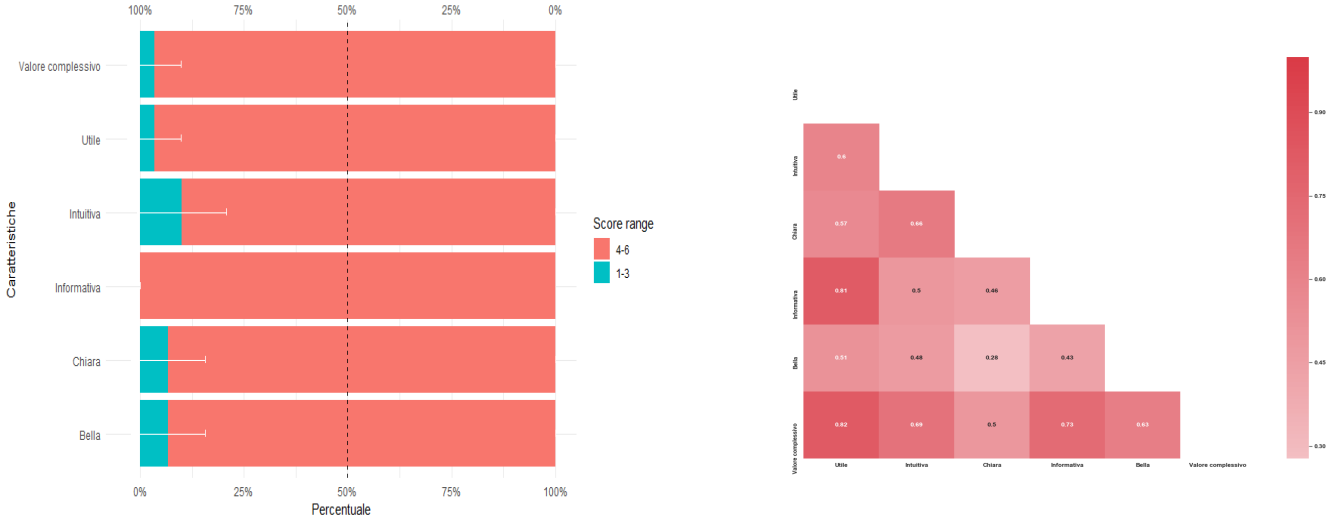




In tutti i campi vi è differenza statisticamente significativa tra opinioni positive e negative, con netta prevalenza per quelle positive. In questo caso, vi sono delle correlazioni abbastanza rilevanti nei seguenti campi:

- tra "Chiara" e "Intuitiva" il coefficiente di correlazione è pari a 0.85
- tra "Chiara" e "Valore Complessivo" il coefficiente di correlazione è pari a 0.82

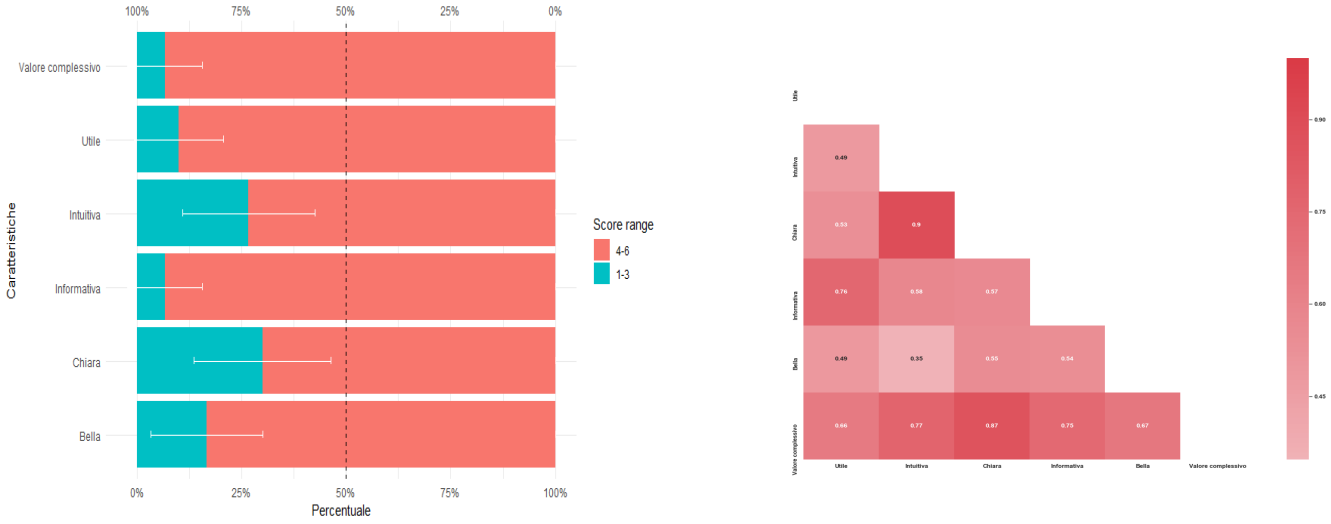
Infografica 3



In tutti i campi vi è differenza statisticamente significativa tra opinioni positive e negative, con netta prevalenza per quelle positive. In questo caso, vi sono delle correlazioni abbastanza rilevanti nei seguenti campi:

- tra "Informativa" e "Utile" il coefficiente di correlazione è pari a 0.81
- tra "Valore complessivo" e "Utile" il coefficiente di correlazione è pari a 0.82

Infografica 4



In tutti i campi vi è differenza statisticamente significativa tra opinioni positive e negative, con netta prevalenza per quelle positive.

In questo caso, vi sono delle correlazioni abbastanza rilevanti nei seguenti campi:

- tra "Chiara" e "Intuitiva" il coefficiente di correlazione è pari a 0.9
- tra "Chiara" e "Valore Complessivo" il coefficiente di correlazione è pari a 0.87

7 Conclusioni

Dallo studio effettuato sui ritardi, risulta evidente che ci sono stati dei picchi prevalentemente nelle regioni del sud Italia, le quali sono anche caratterizzate dalla presenza di un numero minore di treni. Al contrario, al Nord, dove il numero di treni che circolano risulta essere maggiore, il ritardo risulta essere meno evidente. Tuttavia, quasi tutti i treni presentano un ritardo medio di circa 5 minuti, soprattutto nelle fasce orarie tipiche dei pendolari: è evidente che, da questo punto di vista, la rete ferroviaria italiana ha bisogno di essere migliorata.

Oltre ai risultati negativi evidenziati dalla nostra analisi sui ritardi, abbiamo riscontrato delle problematiche in fase di raccolta e pulizia dei dati. Queste ci suggeriscono la presenza di problemi dal punto di vista logistico e di rilevazione dei dati.

In conclusione, come già accennato precedentemente, grazie alle scelte progettuali da noi effettuate, si potrebbe scalare il lavoro compiuto su un periodo temporale maggiore, in modo da approfondire diverse tematiche. Da una parte si potrebbe entrare più nel dettaglio delle analisi da noi compiute, grazie alla possibilità di confrontare una mole maggiore di dati. Dall'altra, si aprirebbero le porte a tipi differenti di analisi che consentirebbero, ad esempio, l'uso di modelli predittivi al fine di migliorare la qualità del servizio.

Riferimenti bibliografici

[1] https://www.stradeeautostrade.it/notizie/2018/istat-in-italia-ogni-giorno-30-milioni-di-pendolari/

[2] https://github.com/bluviolin/TrainMonitor/wiki/API-del-sistema-Viaggiatreno

[3] https://www.ilsole24ore.com/art/notizie/2019-02-10/treni-ad-alta-velocita-18mila-ore-ritardo-2018-1
shtml?uuid=AFRcXjM

[4] https://github.com/sabas/trenitalia

[5] https://www.amcharts.com

[6] https://docs.mongodb.com/manual/

[7] https://neo4j.com/docs/

[8] https://geopy.readthedocs.io/en/stable/

[9] https://www.rdocumentation.org/packages/ggplot2/versions/3.1.1