I consumi energetici delle famiglie italiane nel 2021

a cura degli studenti:
Francesco Paolo Rosone, Salvatore Viganò

Sommario

Idea del progetto

L'idea di questa tesina nasce dalla possibilità di poter capire quali e quante regioni d'Italia ad oggi hanno i più alti consumi energetici a carico delle famiglie nel 2021. Il dataset che abbiamo a disposizione, infatti, contiene le percentuali di famiglie delle regioni italiane che hanno in dotazione uno di questi servizi: riscaldamento dell'abitazione, acqua calda e condizionamento.

Gli obiettivi che ci siamo posti sono i seguenti:

1. Determinare quali sono le regioni italiane con più alto consumo di dotazioni (che sia riscaldamento, acqua calda o condizionamento);
2. Determinare quali sono le regioni italiane con i consumi più alti riguardo ai combustibili utilizzati per l'acqua calda;
3. Determinare, durante la giornata, quali sono le regioni italiane che utilizzano gli impianti di riscaldamento.

Dataset utilizzato

Il dataset che abbiamo deciso di utilizzare si chiama "Indagine Istat sui Consumi Energetici delle Famiglie (PSN IST-02514) - Anno 2021", proveniente direttamente dal sito dell'ISTAT. Inoltre, la licenza supportata a questo dataset è del tipo *Creative Commons (CC BY 3.0 IT)*

Fonte: https://www.istat.it/it/archivio/272110

Procedimento

Manipolazione dei dataset

Una volta scaricato il dataset (disponibile solamente in formato .xlsx) e posizionato all'interno della cartella chiamata DatasetPartenza, la prima fase del progetto è sicuramente manipolare il dataset in modo tale da poter avere dei file .csv utilizzabili per i nostri scopi. Creiamo quattro funzioni dedicate proprio a questa fase, chiamate *datasetIniziale*, *datasetAcquaCalda*, *datasetCondizionamento* e *datasetRiscaldamento*:

- Nella funzione *datasetIniziale*, tramite l'utilizzo della libreria pandas, leggiamo il dataset originale tramite la funzione *read_excel*. Successivamente:
    - Rinominiamo i nomi delle regioni che potrebbero essere scritti con eventuali caratteri speciali;
    - Eliminiamo le colonne che potrebbero risultare vuote o superflue;
    - Rimpiazziamo valori che potrebbero non essere disponibili in un formato a noi standard (vedasi leggenda presente all'interno del dataset originale).

  Una volta che abbiamo ottenuto questo pseudo dataset, abbiamo rinominato i nomi delle colonne in quanto risultano essere diverse da quelle previste (abbiamo stampato l'header del dataset originale in modo da capire quali sono i nomi delle colonne). Il dataset finale che abbiamo ottenuto prende il nome di *datasetIniziale.csv*, presente all'interno della cartella Dataset aggiornato.

Replichiamo questi cambiamenti che abbiamo effettuato nella funzione *datasetIniziale* anche nei seguenti dataset:

- *datasetAcquaCalda*;
- *datasetCondizionamento*;
- *datasetRiscaldamento*.

Una volta ottenuti questi dataset, ognuno separato da un proprio file .csv, il nostro compito è di determinare quali sono le regioni (nel nostro caso le top 3) che hanno una percentuale di dotazione più alta; per farlo, abbiamo deciso di calcolare la media tramite la funzione *mean* e di stamparle su schermo in modo da ottenere gli indici e, di conseguenza, di capire quali sono le regioni interessate. Creiamo due funzioni apposite per queste informazioni che vogliamo ottenere:

- *valoreMassimo*: utilizziamo *pandas* per leggere il .csv del dataset iniziale (presente nella cartella Dataset aggiornato, identificando le righe con i valori più alti delle tre colonne. Successivamente usiamo la libreria *matplotlib* per stampare un grafico a barre con i valori massimi ottenuti e le loro quantità;
- *valoreMinimo*: l'opposto di *valoreMassimo* dove ricerchiamo i valori minimi per trovare le regioni che hanno una percentuale di dotazioni più bassa.

Le regioni che hanno un'elevata percentuale di dotazioni all'interno di un'abitazione sono: *Veneto, Emilia-Romagna, Sardegna*; invece, le regioni che hanno una bassa percentuale di dotazioni all'interno di un'abitazione sono: *Trentino-Alto Adige, Bolzano e Valle d'Aosta.*

Una volta ottenute le informazioni dalle funzioni *valoreMassimo* e *valoreMinimo*, decidiamo di concatenare i due .csv in un unico .csv in modo tale da poter lavorare più facilmente con le successive fasi. Creiamo, a questo punto, una funzione chiamata *concatenaMaxMinValori*, dove utilizziamo la funzione *concat* presente nella libreria *pandas* per concatenare i dataset (concateniamo le righe, in quanto le colonne in entrambi i csv sono uguali → axis=0). Il file .csv che abbiamo ottenuto lo abbiamo chiamato *Regioni_concatenate*, ed è presente all'interno della cartella Dataset aggiornato/datasetLeggibile.

Dopo aver ottenuto le regioni nella fase precedente, adesso ci occupiamo di:

- Concatenare il dataset dell'acqua calda, presente all'interno della cartella Dataset aggiornato, con il dataset *Regioni_concatenate* ottenuto in precedenza;
- Concatenare il dataset del riscaldamento, presente all'interno della cartella Dataset aggiornato, con il dataset *Regioni_concatenate* ottenuto in precedenza;
- Concatenare il dataset del condizionamento, presente all'interno della cartella Dataset aggiornato, con il dataset *Regioni_concatenate* ottenuto in precedenza;

Una volta ottenuti i nuovi .csv, memorizzati all'interno della cartella Dataset aggiornato/datasetLeggibile, decidiamo di sfruttare i dati ottenuti e di ottenere delle informazioni a livello grafico; in particolare, abbiamo deciso di utilizzare la libreria *matplotlib* per poter generare dei grafici a barre verticali.

Da qui il processo si suddivide in tre funzioni differenti:

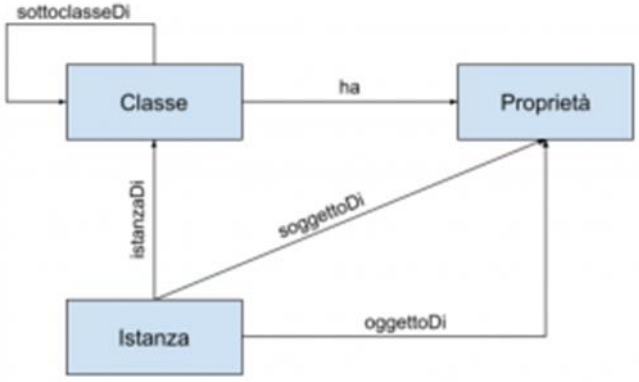
- *datasetAcquaCalda_grafici*: ci permette di capire quali sono le:
  - Le tre regioni che utilizzano più Metano;
  - Le tre regioni che utilizzano meno Metano;
  - Le tre regioni che utilizzano più Energia Solare;
  - Le tre regioni che utilizzano meno Energia Solare.
- *datasetCondizionamento_grafici*: ci permette di capire quali sono le regioni che consumano il sistema di condizionamento nei seguenti periodi:
  - Tutti i giorni o quasi (rinominato come *Giornalmente*);
  - Almeno una volta a settimana (rinominato come *Spesso*);
  - Almeno una volta al mese (rinominato come *Raramente*);
  - Occasionalmente o mai.
- *datasetRiscaldamento_grafici*: ci permette di capire quali sono le regioni che consumano il sistema di riscaldamento nei seguenti periodi:
  - Mattina;
  - Pomeriggio;
  - Sera;

In tutte e tre le funzioni abbiamo generato delle immagini .png dei grafici ottenuti.

Utilizzo di Protégé

Una volta ottenute le informazioni necessarie nel procedimento precedente, abbiamo deciso di creare un'ontologia OWL in modo da poterci riferire ai dati ottenuti. Per ontologia si intende un **modello concettuale che descrive** in modo strutturato e gerarchico i **concetti di un dominio specifico della conoscenza** e le relazioni che definiscono i rapporti tra le diverse entità. Gli elementi principali di un'ontologia sono le:

- **Le classi**: rappresentano i concetti generali del dominio di interesse da rappresentare;
- **Le proprietà**: definiscono il tipo di relazioni che intercorrono tra le classi;
- **Le istanze**: rappresentano oggetti del mondo reale che fanno parte di una determinata classe.



Al fine di **favorire l'interoperabilità semantica** dei dati, sono state utilizzate proprietà di altre ontologie standard esistenti come schema.org, RDFS, OWL, FOAF.

Inoltre, sono state inserite alcune **proprietà custom** per consentire la **descrizione di dati specifici del nostro catalogo** che non potevano essere rappresentati in modo efficace utilizzando le ontologie esistenti (come nel nostro caso). Le ontologie sono state generate in quattro tipologie differenti:

- JSON;
- OWL;
- RDF;
- Turtle.

Per creare queste ontologie personalizzate abbiamo utilizzato il software *Protégé,* un editor di ontologia gratuito e open source che supporta l'ultimo standard OWL 2.0.

Conversione CSV in JSON

Concludendo il nostro progetto, l'ultimo passaggio necessario per far sì che avessimo tutto a portata di mano era di convertire tutti i .csv utilizzati in .json. Per farlo, abbiamo creato una funzione chiamata *conversioneToJSON()* dove, attraverso le librerie *csv* e *json*, abbiamo utilizzato un csvReader (in particolare *csv.DictReader()*) e per ogni riga ottenuta verrà inserita ogni coppia chiave valore nel json con il metodo *append*.