Documentazione

In questo file verranno tratta alcuni aspetti del progetto. Questi aspetti variano dalla descrizione dei dataframe analizzati, a come si sono svolte le analisi statistiche e i loro risultati ad, infine, quali scelte sono state fatte per rappresentare al meglio i dati nei grafici. Il testo è scritto in modo molto informale e non vuole svolgere il compito di una relazione, ma bensì di semplice spiegazione del flusso di idee. Avviso inoltre che non avrò il tempo di revisionare il testo, e nella parte dedicata ai grafici le citazioni alle figure e ai paragrafi potrebbe essere un po' "inconsistenti".

1. Dataframe Analizzati

Nel progetto sono stati analizzati 2 dataframe: uno contenente i dati mensili riguardanti la produzione netta di energia elettrica per tipo di fonte energetica degli stati membri dell'UE (e anche di stati non membri ma comunque affiliati in qualche modo all'unione), l'altro contente i dati annuali sul consumo di energia elettrica nell'UE.

1.1 Net electricity generation by type of fuel

URL: https://ec.europa.eu/eurostat/databrowser/product/view/nrg_cb_pem?category=nrg.nrg_quant.nrg_quantm.nrg_cb_m

1.1.1 Presentazione

Il primo database analizzato è il database sulla produzione netta di energia di 40 nazioni europee (+1 che è il totale degli stati membri, EU27) facenti parte, o in qualche modo affiliate, all'Unione Europea. Le nazioni sono state poi filtrate per includere solo gli stati membri. L'aggiunta di tutti gli stati è stata provata, ma poi successivamente scartata. Bisogna precisare che questo filtro vale per tutta la applicazione. Dunque è stato escluso il Regno Unito (anche nel periodo in cui era ancora membro) e sono state incluse tutte quelle nazioni che sono diventate membri anche successivamente alla data in analisi. Gli stati analizzati sono rappresentati nella seguente tabella:

	Stato	ISO alpha-2			Stato	ISO alpha-2
	Belgio	BE			Lituania	LT

	Stato	ISO alpha-2			Stato	ISO alpha-2
	Bulgaria	BG			Lussemburgo	LU
	Repubblica Ceca	CZ			Ungheria	HU
	Danimarca	DK			Malta	MT
	Germania	DE			Paesi Bassi	NL
	Estonia	EE			Austria	AT
	Irlanda	IE			Polonia	PL
	Grecia	EL/GR*			Portogallo	PT
	Spagna	ES			Romania	RO
	Francia	FR			Slovenia	SI
	Croazia	HR			Slovacchia	SK
	Italia	IT			Finlandia	FI
	Cipro	CY			Svezia	SE
	Lettonia	LV			Unione Europea	EU27_2020

Tabella 1. Nazioni in analisi con il loro codice ISO 3166-1 alpha-2 e la loro bandiera

> **Nota sulla grecia**
>
> La Grecia ha ufficialmente come ISO alpha-2 la siglia GR, ma nel dataframe è indicata come EL, per evitare ambiguità è stato preferito usare il codice ufficiale.

1.1.2 Variabili e Algoritmo

Il dataframe originale oltre all'identificativo degli stati conteneva le seguenti variabili:

- `freq` : frequenza dei dati. Una sola modalità: mensile;
- `time` : mese di riferimento per ciascun dato;
- `siec` : tipo di fonte da cui è stata generata l'energia elettrica;
- `geo` : codice ISO 3166-1 alpha-2 dello stato;
- `unit` : unità di misura. Due modalità: GWH (*gigawattora*) e Percentage.

Successivamente i dati sono stati resi tidy tramite l'algoritmo contenuto nella funzione

`get_data_productivity` nel file `webapp.py`. Nel processo il dataframe è stato modificato:

- La variabile `freq` è stata rimossa perché considerata poco utile;
- `unit` è stata rimossa, ma non prima di filtrare tutti i dati per GWH (gigawattora), eliminando così l'altra unità di misura: *Percentage*;
- `siec` *(Standard international energy product classification)* è stato semplificato rimuovendo tutte quei tipi di fonti di energia considerate troppo specifiche. La seguente lista mostra quali fonti sono rimaste:
  - `TOTAL Total`
  - `CF Combustible fuels`
  - `C0000 Coal and manufactured gases`
  - `G3000 Natural gas`
  - `O4000XBIO Oil and petroleum products (excluding biofuel portion)`
  - `CF_NR Combustible fuels - non-renewable`
  - `RA000 Renewables and biofuels`
  - `RA100 Hydro`
  - `RA200 Geothermal`
  - `RA300 Wind`
  - `RA400 Solar`
  - `RA500_5160 Other renewable energies`
  - `CF_R Combustible fuels - renewable`
  - `N9000 Nuclear fuels and other fuels n.e.c.`
  - `X9900 Other fuels n.e.c.`

    Si potrebbe fare un altro tipo di raggruppamento distinguendo anche le macrocategorie:

    `TOTAL Total`
  - `CF Combustible fuels`
    - `C0000 Coal and manufactured gases`
    - `G3000 Natural gas`
    - `O4000XBIO Oil and petroleum products (excluding biofuel portion)`
    - `CF_NR Combustible fuels - non-renewable`
  - `RA000 Renewables and biofuels`
    - `RA100 Hydro`
    - `RA200 Geothermal`
    - `RA300 Wind`
    - `RA400 Solar`
    - `RA500_5160 Other renewable energies`
    - `CF_R Combustible fuels - renewable`
  - `N9000 Nuclear fuels and other fuels n.e.c.`
  - `X9900 Other fuels n.e.c.`

- Le variabili `time` (rappresentante il mese) e `geo` (il codice ISO della nazione) sono rimaste invariate. Hanno solo cambiato nome diventando rispettivamente: `date` e `state`
- È stata anche aggiunta la variabile `unique_id` composta da `state` e da `siec`, Nel formato: state;siec. Questa variabile è unica per ogni stato e siec e permette un più facile riconoscimento di ogni riga. Tornerà utile per il forecast in quanto così permette di identificare più facilmente sia lo stato che il tipo di energia prodotta;
- `energy_prod`: Variabile continua contenente la quantità di energia elettrica prodotta. Misurata in *GHW*.

```
def get_data_productivity(url):
    data = (
        pl.read_csv(
            url,
            separator="\t",
            null_values=["", ":", ": "],
            )
        .select(
            pl.col("freq,siec,unit,geo\\TIME_PERIOD")
            .str.split(",")
            .list.to_struct(fields=["freq","siec", "unit", "state"])
            .alias("combined_info"),
            pl.col("*").exclude("freq,siec,unit,geo\\TIME_PERIOD")
        )
        .unnest("combined_info")
        .unpivot(
            index=["freq","siec", "unit", "state"],
            value_name="energy_prod",
            variable_name="date"
        )
        .with_columns(
            date=pl.col("date")
            .str.replace(" ", "")
            .str.to_date(format='%Y-%m', strict=False),#, strict=True),
            energy_prod=pl.col("energy_prod")
            .str.replace(" ", "")
            .str.replace("p", "")
            .str.replace("u", "")
            .str.replace("e", "")
            .str.replace("n", "")
            .str.replace("d", "")
            .str.replace(":c", "123456789")
            .cast(pl.Float64),
```

```
            unique_id=pl.col("state")+";"+pl.col("siec"),
            state = pl.col("state")
            .str.replace("EL", "GR")
        )
        .filter(
            pl.col("energy_prod") > 0,
            pl.col("energy_prod").is_not_null(),
            pl.col("energy_prod") != 123456789,
            pl.col("unit") == "GWH",
            pl.col("siec").is_in(["TOTAL","X9900","RA000","N9000",
                                  "CF","CF_R","RA100","RA200",
                                  "RA300","RA400","RA500_5160",
                                  "C0000","CF_NR","G3000","O4000XBIO"]),
        .drop("freq",
              "unit")
        .sort("state", "date")
        )
    return data
```

Codice 1. Funzione `get_data_productivity` del programma `webapp.py`

1.2 Supply, transformation and consumption of electricity

URL: https://ec.europa.eu/eurostat/databrowser/view/nrg_cb_e/default/table?lang=en&category=nrg.nrg_quant.nrg_quanta.nrg_cb

1.2.1 Presentazione

Il secondo database analizzato è sul consumo di energia elettrica di 40 nazioni europee facenti parte, o in qualche modo affiliate, all'Unione Europea. La sua composizione è molto simile a quella del dataframe sulla produzione di energia elettrica. La differenza sostanziale sta nel fatto che mentre nel primo la frequenza è mensile, questo dataframe ha una frequenza annuale. Inoltre, in aggiunta al consumo totale di tutti gli stati membri membri dell'UE, è presente anche quello dei 20 stati appartenenti all'Euro Area. Nonostante questo, il dataframe è stato filtrato allo stesso modo di quello precedente, così da poter permettere il confronto tra essi. gli stati sono dunque gli stessi della *tabella 1*.

1.2.2 Variabili e Algoritmo

Il dataframe originale oltre all'identificativo degli stati conteneva le seguenti variabili:

- `nrg_bal`: 60 tipi diversi di bilancio energetico, tra i più vari come: trasformazione di energia elettrica, uso e consumo in vari settori, import ed export, perdite;
- `freq`: frequenza dei dati. Una sola modalità: annuale;
- `time`: anno di riferimento per ciascun dato;
- `siec`: tipo di energia, una sola modalità: `E7000 Electricity`;
- `geo`: codice ISO 3166-1 alpha-2 dello stato;
- `unit`: unità di misura. Una sola modalità: modalità: GWH (*gigawattora*). Successivamente i dati sono stati resi tidy tramite l'algoritmo contenuto nella funzione `get_data_consumption` nel file `webapp.py`. Nel processo il dataframe è stato modificato:
- Le variabili `freq`, `siec` e `unit` sono state rimosse perché presentavano una sola modalità e sono state considerate poco utili;
- Le variabili `time` e `geo` sono diventate rispettivamente: `date` e `state`
- È stata anche aggiunta la variabile `unique_id` che svolge lo stesso compito della sua controparte nel primo dataframe. L'unica differenza è che è composta da `state` e da `nrg_bal`;
- `energy_cons`: Variabile continua contenente la quantità di energia elettrica prodotta. Misurata in *GHW*.
- La variabile `nrg_bal` è stata filtrata e ora contiene solo i tipi di consumo di energia a noi necessari:
  - `FC Final consumption`
  - `FC_IND_E Final consumption - industry sector`
  - `FC_TRA_E Final consumption - transport sector`
  - `FC_OTH_CP_E Final consumption - commercial and public services`
  - `FC_OTH_HH_E Final consumption - households`
  - `FC_OTH_AF_E Final consumption - agriculture and forestry`

```
def get_data_consumption(url):
    data = (
        pl.read_csv(
            url,
            separator="\t",
            null_values=["", ":", ": "],
            )
        .select(
            pl.col("freq,nrg_bal,siec,unit,geo\\TIME_PERIOD")
            .str.split(",")
            .list.to_struct(fields=["freq","nrg_bal","siec", "unit",
"state"])
            .alias("combined_info"),
            pl.col("*").exclude("freq,nrg_bal,siec,unit,geo\\TIME_PERIOD")
```

```
        )
        .unnest("combined_info")
        .unpivot(
            index=["freq","nrg_bal","siec", "unit", "state"],
            value_name="energy_cons",
            variable_name="date"
        ).with_columns(
            date=pl.col("date")
            .str.replace(" ", "")
            .str.strptime(pl.Date, '%Y'),#, strict=False).cast(pl.Date),
            energy_cons=pl.col("energy_cons")
            .str.replace(" ", "")
            .str.replace("p", "")
            .str.replace("u", "")
            .str.replace("e", "")
            .str.replace("n", "")
            .str.replace("d", "")
            .str.replace(":", "123456789")
            .cast(pl.Float64),
            unique_id=pl.col("state")+";"+pl.col("nrg_bal"),
            state = pl.col("state")
            .str.replace("EL", "GR")
        )
        .filter(
            pl.col("energy_cons").is_not_null(),
            pl.col("energy_cons") != 123456789,
            pl.col("state")!="EA20",
           pl.col("nrg_bal").is_in(["FC", "FC_IND_E" , "FC_TRA_E",
                                    "FC_OTH_CP_E", "FC_OTH_HH_E",
"FC_OTH_AF_E"]),
        )
        .drop("freq",
              "unit",
              "siec")
        .sort("state", "date")
    )
    return data
```

Codice 2. Funzione `get_data_consumption` del programma `webapp.py`

2. Funzione Arima

Per conseguire l'obbiettivo di studiare il deficit/surplus energetico anche per date future, è stato necessario implementare un modello per l'analisi di serie storiche. Vari modelli sono stati provati (principalmente quelli legati alla libraria sktime) come ad esempio: il lisciamento esponenziale, il BATS (Exponential Smoothing+ Box-Cox Transformation + ARMA model for residuals) e TBATS (BATS + Trigonometric Seasonal), Facebook Prophet e ARIMA. Tra tutti questi modelli proprio questo ultimo è risultato il migliore, sia in termini di accuratezza che di facilità di implementazione.
In particolare è stato usato AutoARIMA che include alla stima del modello anche il processo di selezione dei parametri, dettaglio che torna molto utile quando ci si ritrova a dover modellare un numero elevato di serie storiche come in questo caso.

```
[['Naive method', 8, ['UA', 'HU', 'IE', 'EL', 'MD', 'LU', 'TR',
'EU27_2020']],
['AutoARIMA', 32, ['IT', 'ME', 'PT', 'HR', 'SE', 'ES', 'CY', 'NO', 'BE',
'SI', 'BG', 'MT', 'MK', 'NL', 'IS', 'GE', 'EE', 'RO', 'LT', 'SK', 'BA',
'RS', 'AL', 'CZ', 'LV', 'PL', 'DE', 'FI', 'FR', 'UK', 'DK', 'AT'],
['ExponentialSmoothing', 0, [], []],
['AutoETS', 0, [], []],
['BATS', 0, []],
['TBATS', 0, []],
['Prophet', 0, []],
['UnobservedComponents', 0, []],
['Errore', 0, ['XK']
```

Tabella 2. Risultati del confronto tra i MAPE *(mean absolute percentage error)* di vari modelli della libreria sktime di serie storiche stimati sul *database Net electricity generation by type of fuel*. I vari modelli sono stati stimati singolarmente e poi è stato confrontato il loro MAPE, inserendo successivamente il nome dello stato in una lista. È stato scelto il MAPE per la sua facilità di lettura in quanto ha un limite inferiore a 0. Più il modello si avvicina a 0, più si adatta bene ai dati. Si può notare come nella maggioranza dei casi il modello AutoArima risulti il migliore.

```
from sktime.forecasting.bats import MODELLOX
def forecast(y):
    try:
        y_train, y_test = temporal_train_test_split(y, test_size=36)

        fh = ForecastingHorizon(y_test.index, is_relative=False)

        forecaster = MODELLOX(sp=12, use_trend=True, use_box_cox=False)
        forecaster.fit(y_train['energy_productivity'])

        y_pred = forecaster.predict(fh)
```

```
        mape = mean_absolute_percentage_error(
                        y_test['energy_productivity'], y_pred)
        return mape
    except Exception as e:
        print(f"Errore: {e}")
        return None
```

Codice 2. Funzione usata per calcolare il mape per i vari modelli della funzione sktime. Con MODELLOX nome fittizio del vero modello importato.

Scelto il modello, sono state testate due funzioni AutoArima: quella della libreria di Sktime (che è solo una implementazione della omonima classe di pmdarima al fine di renderla compatibile con l'interfaccia di sktime) e infine l'AutoArima del pacchetto statsforecast di Nixtla. Le librerie in termini di accuratezza sono simili, ma il guadagno maggiore lo si ottiene dal punto della velocità di calcolo. Infatti, dato il dataframe sulla produzione di energia, statsforecast esegue i forecast di tutte le fonti di energia di una nazione in 4 minuti, mentre fissata la fonte, per eseguire i forecast di tutte le nazioni ci impiega 2 minuti per una media di circa 7 secondi a forecast. I tempi di sktime/pmdarima sono invece molto più lunghi. Per stimare il modello di tutte le nazioni fissata una fonte AutoARIMA di sktime/pmdrima impiega tra i anche 10 e i 15 minuti (contro i 2 minuti di statsforecast).

	Mape StatsForecast**	**Mape SkTime****
Belgio	0.2083*	0.133
Francia	0.08579	0.137
Germania	0.08441	NA
Italia	0.05927	0.0666
Spagna	0.05844	0.0481

Tabella 3. RIsultati del train del modello, MAPE a confronto.

Nota sul belgio*

Con il normale split tra train e test fissato al 80%, Il Belgio raggiunge la media e ha MAPE uguale a 0.21. Se si ingrandisce di poco la grandezza del train (tipo 85%) il Belgio non raggiunge la media è a MAPE uguale a 0.0801

Nota sulla grandezza del train**

Ho appena notato che la grandezza del databse di train di StatsForecast è llegermente più grande di quella usata originariamente per SkTime.

Nota sulla qualità dei dati

Purtroppo questa parte di raccolta dei valori del MAPE è stata molto scaglionata. Ad inizio novembre ho realizzato il codice da cui ho ottenuto il MAPE di sktime, a metà novembre ho realizzato il codice da cui ho ottenuto il MAPE di StatsForecast, e un mese dopo scrivo il commento, non posso quindi ricordarmi tutti i dettagli di cosa feci. I dati sono giusti, ma qualche informazione potrebbe essere dunque incerta.

```
def Arima_prod(state):
    ts = df_prod.filter(
        pl.col("unique_id") == state
        ).select(
            pl.col("date").alias("ds"),
            pl.col("energy_prod").alias("y"),
            pl.col("unique_id")
            )
    len_date = len(ts.select("ds").unique())
    perc = round((len_date*80)/100)
    split = ts.select("ds").unique().sort("ds").to_series()[perc]
    Y_train_df = ts.filter(pl.col("ds")<= split)
    Y_test_df = ts.filter(pl.col("ds")> split )

    sf = StatsForecast(
        models = [AutoARIMA(season_length = 12)],
        freq = '1mo',
        n_jobs=-1,
        fallback_model=SeasonalNaive(season_length=12)
        )
    ts_pred = sf.forecast(df=Y_train_df, h=48, level=[95]) #h = 48
    # sf.fit(df=Y_train_df)
    ts_pred = ts_pred\
        .rename({"AutoARIMA": "energy_prod"})\
        .rename({"AutoARIMA-lo-95": "AutoARIMA_low90"})\
        .rename({"AutoARIMA-hi-95": "AutoARIMA_hi90"})
    Y_hat_df = sf.forecast(df=Y_train_df, h=48, fitted=True) #h = 48
    # Y_test_df = Y_test_df.to_pandas()
    # Y_hat_df = Y_hat_df.to_pandas()
    adattamento = evaluate(
        # Y_test_df.merge(Y_hat_df, how='left', on=['unique_id', 'ds']),
        Y_test_df.join(Y_hat_df, how='left', on=['unique_id', 'ds']),
        metrics=[ufl.mae, ufl.mape, partial(ufl.mase, seasonality=12),
ufl.rmse, ufl.smape],train_df=Y_train_df,
```

```
    )


    return ts_pred, adattamento, Y_test_df
```

Codice 3. Funzione usata per il train di AutoARIMA di statsforecast

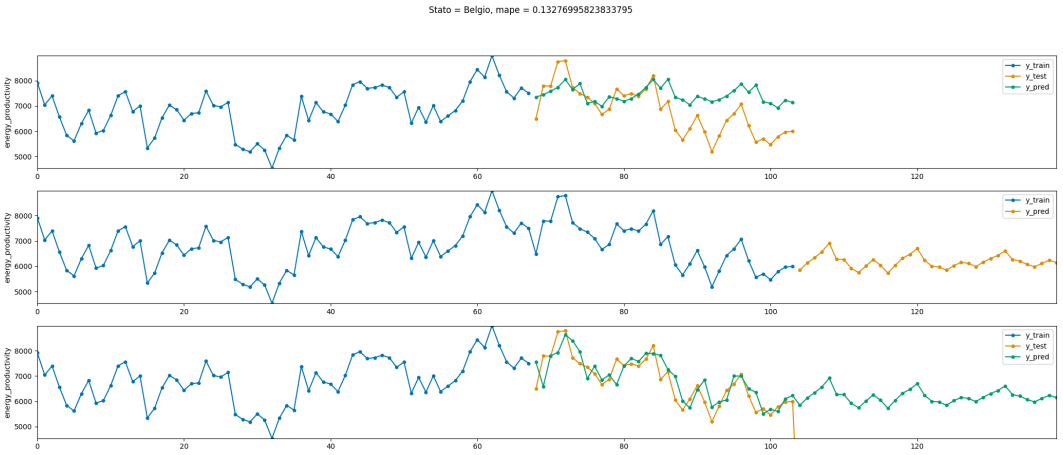

Figura 1. Previsioni AutoARIMA sktime/pmdarima Belgio

Figura 2. Previsioni AutoARIMA statsforecast Belgio

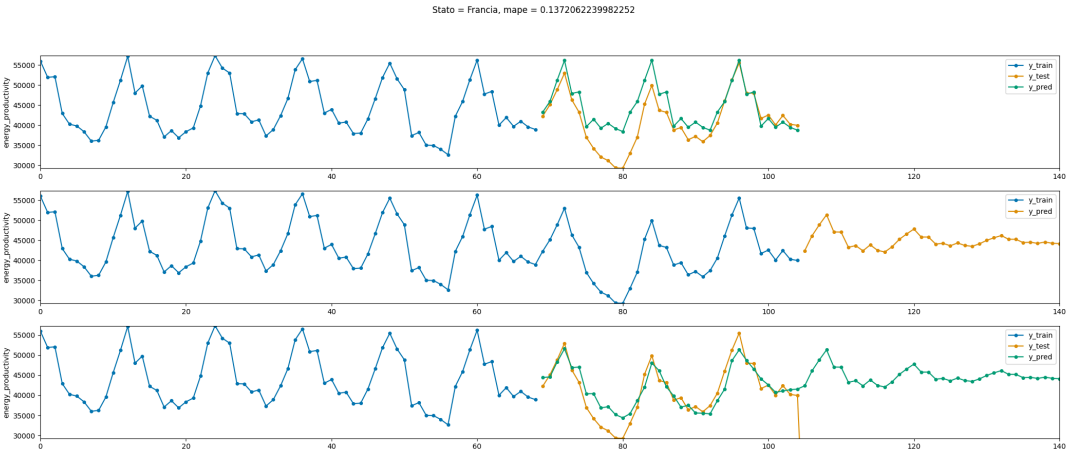

Figura 3. Previsioni AutoARIMA sktime/pmdarima Francia

Figura 4. Previsioni AutoARIMA statsforecast Francia

Figura 4. Previsioni AutoARIMA statsforecast Germania

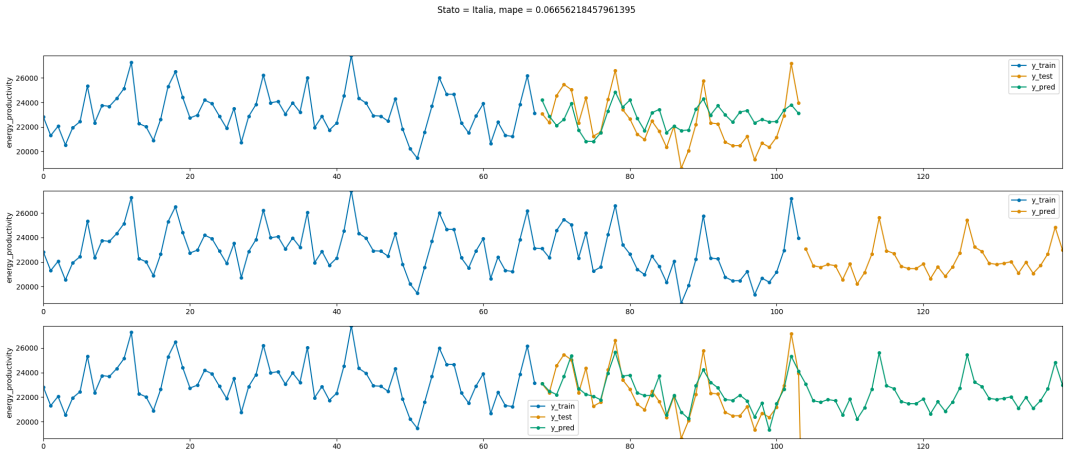

Figura 5. Previsioni AutoARIMA sktime/pmdarima Italia

Figura 6. Previsioni AutoARIMA statsforecast Italia

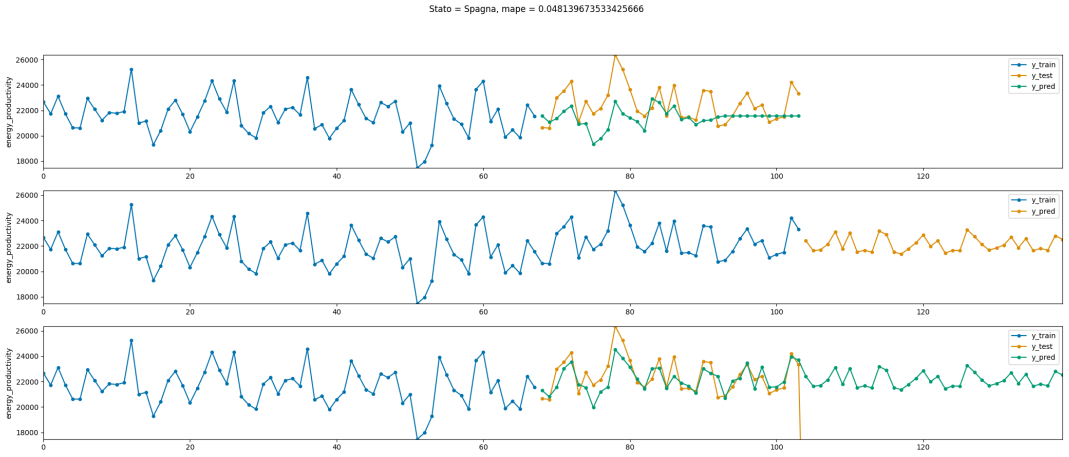

Figura 7. Previsioni AutoARIMA sktime/pmdarima Spagna

Figura 8. Previsioni AutoARIMA statsforecast Spagna

2.1 Problema con i dati del datframe consumo

Sembrava doveroso concentrarsi un attimo sul datframe consumo. Questo dataframe contiene i consumi annuali per i vari stati europei dal 1990 al 2022/2023 (a seconda di chi ha già condiviso i dati). Purtroppo o la scarsa quantità di osservazioni (solo 36), una bassa qualità dei dati ha fatto si che ogni volta che si provasse a fare una previsione con un modello ARIMA questo raggiungesse subito la media. Per risolvere questo problema si è

provato ad implementare una tecnica di upscaling della serie cerncandola di portarla da una frequenza annuale ad una frequenza inferiore. Sono state provate serie trimestrali e quadrimestrali, ma hanno portato scarsi risultati. Un miglioramento di è iniziato ad ottenere con una frequenza bimestrale. Alcuni stati (soprattutto quelli principali) hanno iniziato a mostrare dei miglioramenti. Il tutto però rimane molto fragile tanto che, con l'aggiornarsi del dataframe la serie di EU27 che prima non presentava problemi, è tornata a raggiungere velocemente la media. In ogni caso anche se le nazioni non raggiungono la media le previsioni sono pessime. Tanto pessime che è quasi impossibile provare a replicare ciò che è stato fatto per misurare l'accuratezza delle previsioni perchè riducendo la dimensione del dataframe, anche del minimo, il modello raggiunge immediatamente la media appena esce dal dataset di train. È stato provato acneh ad applicare modelli di serie storiche più semplici, ma più semplice di un ARIMA ho trovato solo i modelli AR o MA che comunque non hanno portato a nessun miglioramento.

```
ts_upscale = (
        ts.upsample(time_column="ds", every="2mo")
        .interpolate()
        .fill_null(strategy="forward")
    )
```

Codice 4. Operazione di upscaling implementata in `Arima_cons`

3. Database Deficit/Surplus

Dopo aver spiegato come si sono volte le analisi, non resta che spiegare come è stato composto il dataframe Deficit. Innanzitutto, a differenza dei due precedenti dataframe analizzati, defiit/surplus non è stato scaricato, ma invece è il prodotto dei due dataframe sulla produzione e sul consumo dei dati. La prima operazione è stata trasformare il dataframe sulla produzione da una frequenza mensile ad una annuale. Successivamente è stato eseguito il join tra i due dataframe ottenendo così un dataframe unico. Per poi infine fare la differenza tra il consumo totale di energia elettrica e la produzione totale di energia elettrica per ogni stato dell'unione europea. L'alternativa era rappresentata dal fare la differenza prima del forecast e poi fare le previsioni sul dataset ottemtuo. Il problema era rappresentato dal fatto che questa opzione si portava indietro lo stesso probleema del dataframe consumo, ovvero la mancanza di tanti dati. Dopo aver constatato questo, si ha optato per l'opzione attuale, ovvero fare la differenza in un momento successivo al forecast dei due dataframe.

Figura 9. Grafico ottenuto dalla differenza tra il dataframe consumo e produzione e successivo forecast sul nuovo dataframe. Si osserva come già al primo dato l'ARIMA ha raggiunto la media.

Figura 10. Grafico ottenuto dalla differenza tra il dataframe consumo e produzione dopo aver ottenuto le previsioni dei singoli database. Si può notare come a differenza della figura precedente questa dimostri un andamento più regolare (ignorando l'anno 2022, anno particolare per l'energia in europa a seguito dell'inizio del conflitto russo-ucraino, in cui si osserva un picco vertigonoso del surplus di energia per tutti gli stati europei)

4. Grafici

Dopo aver spiegato le analisi statistiche dietro al progetto, si passa ora a spiegare le varie scelte di motivi e di stile che hanno portato alla implementazione dei grafici attualmente visualizzati.

4.0.1 Elementi in comune dei grafici

Prima di analizzare i vari singoli grafici sarebbe opportuno elencare alcuni elemneti grafici che li accumunano. In particolare è stata posta molta attenzioni a dividere i valori osservati da quelli predetti tramite ARIMA così da rendere più immediata la lettura del grafico da perte dell'utente. Per fare ciò innanzitutto è stata divisa la finestra grafica in due parti: una parte bianca, contenente i valori osservati, e una parte grigia contente i forecast separate da una linea tratteggiata verticale. Inoltre mentre per i valori osservati è stata usata una linea continua, per le previsioni è stata usata una linea tratteggiata.
Documentazione usata:

- https://altair-viz.github.io/gallery/falkensee.html (per il rettangolo grigio)
- https://altair-viz.github.io/gallery/line_chart_with_datum.html (per la linea verticale)
- https://altair-viz.github.io/gallery/line_with_last_value_labeled.html (per i nomi a fine grafico)
- https://altair-viz.github.io/gallery/multiline_tooltip_standard.html (tooltip a "finestrella")
- https://altair-viz.github.io/gallery/multiline_tooltip.html (tooltip con valori sulla linea)

4.1 Mappa

Il primo dei grafici implementato è una mappa raffigurante o la produzione o il consumo o il deficit-surplus di energia elettrica per gli stati membri dell'unione europea. È possibile, tramite lo slide che si trova sopra cambiare l'anno di cui si vogliono visualizzare i dati. È inoltre l'unico grafico uguale per tutte e tre le pagine. La maggior parte del codice di questo grafico, sopratutto come zoomarlo, è ispirato da una domanda postata su stack overflow (How to Center or Crop a Map of Europe?), mentre il json è stato preso dalla seguente repository github: world-atlas. La scala del json è 1:50 che permette di mantenere comunque una ottima qualità. È stata testata anche la mappa con scala 1:10, ma poi reputata "inutile" in quanto la mappa 1:50 offre di suo una ottima qualità. Lo schema di colori che è stato scelto per rappresentare i dati è la palette "inferno". La scelta è ricaduta su questa palette, e non su altre simili come plasma e viridis, sia per la sua "robustezza" ai vari tipi di daltonismo (caratteristica che condivisa da tutte le palette citate), ma sopratutto per la presenza del colore nero all'estremo superiore della scala che permette meglio di catturare l'attenzione dell'utente facendogli notare più facilmente il dato in analisi che può essere, a seconda di cosa si sta visualizzando, una situazione di elevata produzione di energia elettrica, o di consumo o di surplus . L'unica situazione in cui la scala può creare un po' di confusione potrebbe essere nella pagina *deficit*/surplus in quanto il nero rappresenta le nazioni con un forte surplus. Questa situazione, a seconda se viene data una maggiore importanza al deficit o al consumo, potrebbe risultare, per alcuni un po' ambigua e contro intuitiva, perché ci si

potrebbe aspettare che siano colorati di nero (colore a cui viene data una accezione di solito negativa) gli stati che si trovano, appunto, in situazioni di elevato deficit. L'idea di un cambio di palette (sia in termini di dominio che proprio di colori) è stata provata ma poi bocciata. Ho trovato che invertire il dominio (con il colore nero a rappresentare gli stati in forte deficit e il giallo gli stati in forte surplus) fosse sostanzialmente inutile in quanto si sarebbe trattato solo di spostare il problema, in quanto ora, sarebbero state le nazioni con un forte surplus a diventare in secondo piano. La scelta più ovvia sembrerebbe il cambiamento della palette. Sono state provate varie palette, principalmente con colori divergenti e sia in scala continua che discreta. Purtroppo per vari motivi non è stata trovata una palette soddisfacente. Innanzitutto non trovavo carina l'idea che, mentre due mappe condividessero lo stesso schema colori, la terza ne presentasse uno totalmente divers. L'alternativa principale consisteva nello "spaccare" manualmente la scala allo 0, ma per inesperienza e con la paura di fare danno, questa idea è stata scartata. È stato quindi deciso di mantenere la scala di default, cosciente di tutti i problemi che essa presenta.

4.2 Line Chart pagina Deficit

Il line chart nella pagina deficit è molto standard, ma aiuta a farsi una idea dello stile dei grafici in tutto il progetto. È anche presente una linea tratteggiata sullo zero, che serve a meglio discriminare i valori positivi da quelli negativi. Per una più facile lettura dei dati e un confronto più immediato è stata implementata uno slide verticale che stampa a video una etichetta con il valore. La palette scelta per rappresentare gli stati(anche negli altri grafici) è *tableau10*. È stata scelta in quanto il grafico è pensato per rappresentare solo una manciata di stati alla volta (benché possano essere scelti anche tutti), e questa palette contiene 10 colori tutti ben distinti tra di loro.
Documentazione usata per comporre il grafico:

- https://altair-viz.github.io/gallery/line_chart_with_custom_legend.html
- https://altair-viz.github.io/gallery/multiline_tooltip_standard.html
- https://altair-viz.github.io/gallery/line_chart_with_datum.html

4.3 Bar plot Classifica

Il terzo grafico della pagina deficit mostra in una unica rappresentazione tutte le nazioni europee e la loro performance in termini di deficit o surplus energetico. Il grafico è molto semplice: gli stati sono colorati in modo diverso a seconda che si trovino in una delle due situazioni. Ogni barra ha accanto l'etichetta dello stato e il valore associato. L'anno di cui si visualizza i dati è lo stesso scelto tramite lo slide che si è discusso nel paragrafo dedicato alla mappa. Non è stata implementata una linea verticale sull'asse x=0 (sulla falsariga del line plot precedente)in quanto se fosse stato fatto così altair avrebbe eliminato le attuali righe verticali. È stato quindi scelto di mantenere quest'ultime in quanto permettono in maniera più immediata il confronto tra le varie barre. Se ci si volesse concentrare su una barra in particolare è possibile farlo cliccando sulla barra evidenziata. Il click mette in sovrimpressione la barra e rende meno evidenti tutte le altre. È anche presente la media

europea così da poter fare un più veloce confronto tra uno stato e il resto dell'unione. La palette dei colori è formata da un verde molto scuro e da un bordeaux. Questa palette è anche adatta a persone daltoniche come può essere facile mente notato dalla seguente immagine:

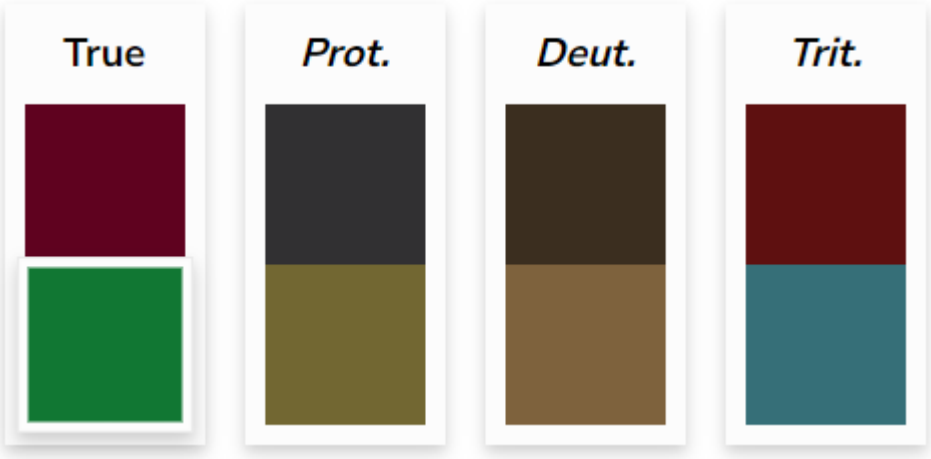

Figura 11. Palette Bar plot
Documentazione usata per comporre il grafico:

- https://altair-viz.github.io/gallery/bar_chart_with_negatives.html
- https://altair-viz.github.io/gallery/bar_chart_with_labels.html
- https://altair-viz.github.io/gallery/interactive_bar_select_highlight.html

4.4 Bump chart classifica

L'ultimo grafico della pagina deficit/surplus è un bump chart che ha lo scopo di evidenziare come si evolve la classifica degli stati con un maggiore surplus di energia. In particolare il grafico mostra le prime 5 posizioni. In quanto le numeroso linee che si intrecciano possono dare problemi alla lettura del grafico, è stato implementato un tooltip verticale che permette di visualizzare il codice degli stati in ordine di classifica. Inoltre, a differenza di quasi tutti gli altri grafici dotati di linee, qui queste ultime perdono la loro funzione di rappresentare i dati osservati da quelli futuri. Questa scelta è stato scelto che le linee tratteggiate alleggerissero il grafico in quanto hanno un alinea meno marcata, inoltre permettono di meglio vedere i punti rappresentanti la posizione dello stato in classifica. La palette usata è *tableau10*, condivisa con tutti i grafici che devono rappresentare nazioni (per approfondimenti vedere *4.2 line Chart deficit*).
Documentazione usata per comporre il grafico:

- https://altair-viz.github.io/gallery/bump_chart.html
- https://altair-viz.github.io/gallery/multiline_tooltip.html

4.5 Lineplot con Tooltip e Doppio Asse

Il secondo dei grafici implementati nella pagina *production* è un line plot con doppio asse e con un tooltip che permette di evidenziare ad un dato tempo la produzione di energia per ogni paese in esame. Il doppio asse è pensato per permettere di rappresentare la media

europea mantenendo la sua scala originale. La media infatti sarà sempre rappresentata (anche se non è stato selezionato nessuno stato) in grigio e con una leggera trasparenza. Il grafico mantiene le caratteristiche grafiche del precedente (linee tratteggiate e non, finestra grafica divisa tra bianco e grigio, per una spiegazione più dettagliata vedere grafico *4.2 Line Chart Deficit*). Come per il line chart del deficit la palette scelta è *tableau10* per la sua caratteristica dei colori accesi e distinti, anche se qui potrebbe andare a creare un po' di confusione al decimo stato visualizzato in quanto anche questo risulterebbe grigio come la media europea. Nonostante questo, e sempre per un discorso simile per il cambio di palette in *3.1 Mappa* è stato scelto di tenere la stessa palette del line chart precedente.
Documentazione usata per comporre il grafico:
- https://altair-viz.github.io/gallery/line_chart_with_custom_legend.html
- https://altair-viz.github.io/gallery/multiline_tooltip_standard.html
- https://altair-viz.github.io/gallery/layered_chart_with_dual_axis.html

4.6 Area Chart

Per rappresentare l'evoluzione della produzione di energia elettrica in un singolo stato è stato scelto un area chart con i vari tipi di produzione di energia sovrapposti. Dato che la visualizzazione contemporanea di tutti i tipi di energia avrebbe portato confusione e reso il grafico illeggibile, è stato scelto di visualizzare solo le macro categorie. In particolare:

- `CF Combustible fuels`
- `RA000 Renewables and biofuels`
- `N9000 Nuclear fuels and other fuels n.e.c.`
- `X9900 Other fuels n.e.c.`
  La palette scelta è ispirata dalla palette Wong, da cui sono stati presi il nero, il verde, il giallo, e il blu che sono serviti a rappresentare i vari tipi di produzione di energia (rispettivamente CF, RA000, N9000, X9900).

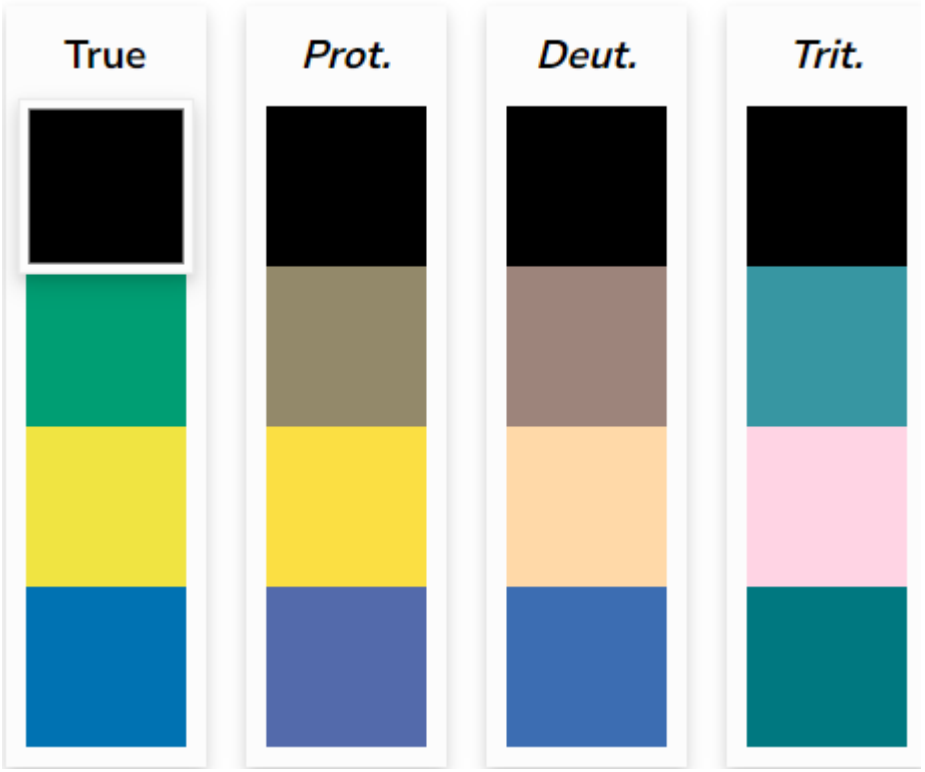

Figura 12. Palette Area Chart

Per rendere ancora più leggibile il grafico è stata scelta l'interpolazione *"step-after"*, che rende il grafico più spigoloso e meno continuo così da riuscire meglio a distinguere i vari passaggi nel tempo. La lettura dei singoli valori è agevolata dalla presenza di un tooltip che crea una piccola finestrella in cui viene visualizzata la sigla della produzione e il valore di energia elettrica prodotta nel rispettivo mese. È facile notare i miglioramenti portati da tutte queste scelte confrontando l'attuale grafico mostrato in output dalla web app con il grafico originale:

Figura 13. Vecchio Area Chart con tutte le fonti di energia, in particolare è rappresentata

Italia
Il grafico tornerà utile per parlare di un altro problema riscontrato causato dalla mancanza di alcuni dati.
Documentazione usata per comporre il grafico:

- https://altair-viz.github.io/gallery/normalized_stacked_area_chart.html
- https://altair-viz.github.io/gallery/filled_step_chart.html

4.7 Bar Chart Produzione + Database

Dopo aver osservato come una singola nazione produce la propria energia elettrica guardando alle macro-categorie, il passaggio successivo non può essere che entrare nel dettaglio. Questo compito è svolto da un bar plot a cui è stato affiancato una tabella contente il totale di energia prodotta per ogni tipo di fonte (vedere sezione *1.1 Dataset Produzione*) dato un anno (selezionato dallo slide ad inizio pagina). L'idea originale consisteva in un grafico più organico ispirato dal seguente:

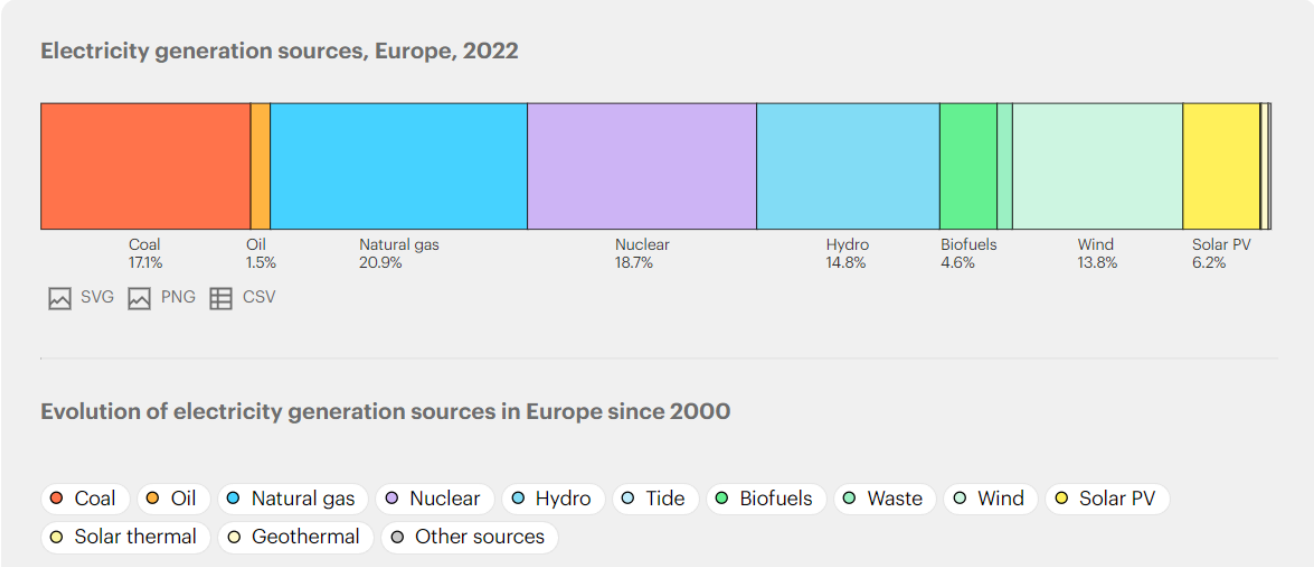

Figura 14. Bar plot sulla produzione di energia dal sito https://www.iea.org/regions/europe/electricity
Ma per limitazioni di Altair non è stato possibile. Il risultato che era stato raggiunto è il seguente:

Figura 15. Barplot e barra inferiore unite usando vconcat di altair (https://altair-viz.github.io/user_guide/compound_charts.html#vertical-concatenation)

che è stato valutato come troppo sconnesso e successivamente sostituito con il grafico attuale. Anche se a prima vista la tabella e il grafico possono sembrare sconnessi, la loro lettura è facilitata dal fatto che le fonti siano entrambe ordinate allo stesso modo.
La palette scelta è *category10*, leggermente diversa da *tableau10* (usata per la rappresentazione degli stati), questa palette risulta buona in quanto ciascun colore risulta essere ben distinto dai due successivi e precedenti. Questa caratteristica, che può sembrare banale, risulta fondamentale quando ci si ritrova a rappresentare percentuali o quantità molto piccole perché spesso ci si ritrova con barre orizzontali così piccole da quasi scomparire. Se, ad esempio, la palette alterasse colori caldi e colori freddi questa potrebbe risultare non adatta in quanto ci si potrebbe ritrovare con due colori dello stesso tono.
Un grosso problema, che ha portato alla aggiunta di un warning, è la assenza di dati in alcune nazioni. Ad esempio, come si può anche vedere dalla *figura 11*, nessuna nazione nell'anno 2017 ha dichiarato la propria produzione totale di energia rinnovabile (motivo per cui tutti i grafici della pagina sono impostate per visualizzare dati dopo il 2017), inoltre per alcuni stati la situazione è ancora più grave. Ad esempio l'Italia non ha dichiarato la sua produzione per le singole fonti non rinnovabili fino al 2022, preferendo dichiarare solo la produzione totale dal non rinnovabile, in *figura 11* si può infatti notare l'assenza di `C0000`, `CF_NR`, `G3000`, `O4000XBIO` e `N9000`). Per sopperire a questa mancanza di stati è stato scelto di aggiungere un warning che avvisa della presenza di dati mancanti (o al contrario anche della troppa presenza di alcuni valori) facendo un confronto tra la somma di tutta la energia elettrica prodotta in un anno (calcolata come somma delle singole quantità) con quella totale reale presente nel data frame come *TOTAL*).
Documentazione usata per comporre il grafico:

- https://altair-viz.github.io/gallery/simple_bar_chart.html
- https://docs.streamlit.io/develop/api-reference/layout/st.columns
- https://altair-viz.github.io/gallery/multiline_highlight.html (per l'highlight delle singole barre)

4.8 Bar Plot Consumo

Come primo grafico specifico della pagina consumo è stato scelto un barplot con lo scopo di confrontare i vari tipi di consumo di energia elettrica pro capite per ogni paese dell'unione. I valori sono confrontati tra di loro e con la media dell'UE (rappresentata dalla linea tratteggiata verticale). L'anno dei dati è scelto tramite lo slide ad inizio pagina. In particola si voleva accentuare tutti quei valori sopra la media. Per fare ciò tutti i valori sotto la media sono stati colorati di grigio, mentre spiccano con un colore rosso tutti questi valori che sforano la soglia. La palette scelta (anche se riguardante solo le etichette) è la usuale *tableau10* (discussa in *3.2 Line Chart deficit*).
Documentazione usata per comporre il grafico:

- https://altair-viz.github.io/gallery/bar_chart_with_single_threshold.html
- https://altair-viz.github.io/gallery/bar_chart_faceted_compact.html

4.9 Line Chart Consumo

Per analizzare l'evoluzione del consumo di energia elettrica negli anni è stato scelto un line chart. Il grafico mantiene tutte le caratteristiche dei precedenti (*4.2 Line Chart Deficit*). Consente un tooltip che permette una veloce lettura del valore di consumo di energia. La sua caratteristica principale consiste nel mostrare gli intervalli di confidenza calcolati insieme alle stime del forecast tramite ARIMA. La scala dei colori scelta è la *category10* (la stessa usata per le fonti di produzione di energia) scelta principalmente purché presenta i primi 5 colori (cioè quelli che a noi interessano perché 5 sono i tipi di consumo in analisi) molto ben distinti tra di loro.
Documentazione usata per comporre il grafico:

- https://altair-viz.github.io/gallery/line_with_ci.html

4.10 Pie Chart Consumo

Infine por dopo aver visto l'evoluzione in termini assoluti del consumo di energia in uno stato europeo, analizziamo quanto ogni singolo tipo di consumo infici, in percentuale, sul consumo totale di energia. Il pie chart visualizzato svolge proprio questo scopo: rappresentare in maniera veloce e immediata la percentuale di consumo energetico dato un certo anno in un paese europeo. Come la controparte nella pagine *Produzione*, anche questo grafico è accompagnato da una tabella.
Documentazione usata per comporre il grafico:

- https://altair-viz.github.io/gallery/donut_chart.html
- https://docs.streamlit.io/develop/api-reference/layout/st.columns
- https://altair-viz.github.io/gallery/pie_chart_with_labels.html