FFHS
Fernfachhochschule Schweiz
Mitglied der SUPSI

Dokumentation & Reflexion

Visualisierung der Verkehrsentwicklung des
Schweizerischen Straßen und Luftverkehrs
Von 1950 - 2019

Josef Gulyas
DaVi, Data Visualization
29.05.2020

Datenquelle

Die Daten zu meiner Semesterarbeit zum Thema „Visualisierung der Verkehrsentwicklung des Schweizerischen Straßen und Luftverkehrs“, habe ich aus der Datenbank vom Bundesamt für Statistik heruntergeladen.
Das Rubrik zum Thema Zivilluftfahrt wird unter folgendem Link erläutert:

- https://www.bfs.admin.ch/bfs/de/home/statistiken/mobilitaet-verkehr/querschnittsthemen/zivilluftfahrt.html

Die Daten für die Flugbewegungen und Flugpassagiere können direkt aus der Datenbank extrahiert werden. Für meine Arbeit habe ich CSV als Export – Format gewählt.

- https://www.pxweb.bfs.admin.ch/pxweb/de/px-x-1107020000_101/px-x-1107020000_101/px-x-1107020000_101.px
- https://www.pxweb.bfs.admin.ch/pxweb/de/px-x-1107020000_102/px-x-1107020000_102/px-x-1107020000_102.px

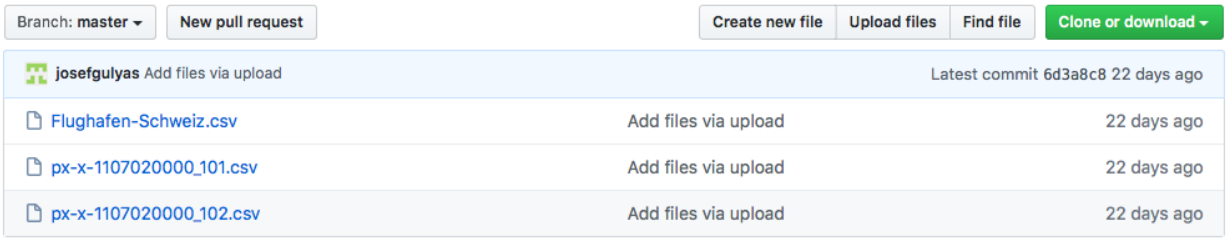
Da ich keine Geo-Daten zu den entsprechenden Flughäfen gefunden habe, musste ich selber eine erstellen. Die Koordinaten habe ich auf folgender Webseite eingegeben und danach in eine Textdatei geladen.

- https://geosearch.mws-design.ch/GeoSearch.html?

Download der Daten

Für die Webapplikation habe ich GitHub als Source verwendet. Die Daten sind zentral an einem Ort abrufbar. Der Link zu den Daten lautet wie folgt:

- https://raw.githubusercontent.com/josefgulyas/ffhs/master/



Beschreibung der Datenfelder

Tabelle:	Datenfeld:	Beschreibung:
Flugbewegungen (px-x-1107020000_101.csv)	Flughafen	Alle Flughäfen der Schweiz gruppiert nach Landesflughäfen (Zürich, Basel und Genf) und nach Regionalflughäfen (Bern, Lugano, Sion und St. Gallen). Zusätzlich sind alle Flughäfen einzeln aufgeführt.
	Art des Verkehrs	Alle Flugbewegungen gruppiert nach Linienverkehr, Charterverkehr und Total.
	Monat	Gruppierung nach Monat sofern erfasst. Diese Gruppierung wurde erst seit Jahr 2000 erfasst.
	Jahr	Gruppierung nach Jahr. Die Jahre zwischen 1950 - 1980 wurden in Zehnjahresschritten, 1980 – 1995 in Fünfjahresschritten und 1995 – 2019 wurden alle Jahre erfasst.
Flugpassagiere (px-x-1107020000_102.csv)	Flughafen	Siehe Flugbewegungen!
	Art des Verkehrs	Siehe Flugbewegungen!
	Art der Passagiere	Gruppierung der Passagiere nach Lokal und Transferpassagiere und Transitpassagiere. • Lokalpassagiere: Passagiere bei welchen die Schweiz das Start/Ziel der Reise ist. • Transit Transferpassagiere: Passagiere bei welchen die Schweiz nicht das Abflug oder Zielland der Reise ist.
	Monat	Siehe Flugbewegungen!
	Jahr	Siehe Flugbewegungen!
Flughafen-Schweiz (Flughafen-Schweiz.csv)	Flughafen	Siehe Flugbewegungen!
	Latitude	Latitude Koordinaten
	Longitude	Longitude Koordinaten

Dateninhalt

Für die Visualisierung wurden folgende Filter angewendet und als CSV heruntergeladen:

Anzahl Flugbewegungen			
Flughafen:	**Art des Verkehrs**:	**Monat**:	**Jahr**:
Schweiz **>> Landesflughäfen** **... Basel-Mulhouse** **... Genève Cointrin** **... Zürich Kloten** **>> Regionalflugplätze** **... Bern-Belp** **... Lugano-Agno** **... Sion** **... St.Gallen-Altenrhein**	Art des Verkehrs – Total Linienverkehr Charterverkehr	Jahrestotal Januar Februar März April Mai Juni Juli August September Oktober November Dezember	1950 1960 1970 1980 1985 1990 1995 1996 1997 1998 1999 2000… 2019

Anzahl Flugpassagiere				
Flughafen:	**Art des Verkehrs**:	**Art der Passagiere:**	**Monat**:	**Jahr**:
Schweiz **>> Landesflughäfen** **... Basel-Mulhouse** **... Genève Cointrin** **... Zürich Kloten** **>> Regionalflugplätze** **... Bern-Belp** **... Lugano-Agno** **... Sion** **... St.Gallen-Altenrhein**	Art des Verkehrs – Total Linienververkehr Charterverkehr	Lokal- und Transferpassagiere Transitpassagiere	Jahrestotal Januar Februar März April Mai Juni Juli August September Oktober November Dezember	1950 1960 1970 1980 1985 1990 1995 1996 1997 1998 1999 2000… 2019

Flughafen-Schweiz		
Flughafen:	**Latitude**:	**Longitude:**
... Basel-Mulhouse	47.598165	7.525485
... Genève Cointrin	46.237010	6.109156
... Zürich Kloten	47.458216	8.555476
... Bern-Belp	46.912222	7.499167
... Lugano-Agno	46.004416	8.910109
... Sion	46.222706	7.339478
... St.Gallen-Altenrhein	47.485874	9.560018

Visualisierung der Daten

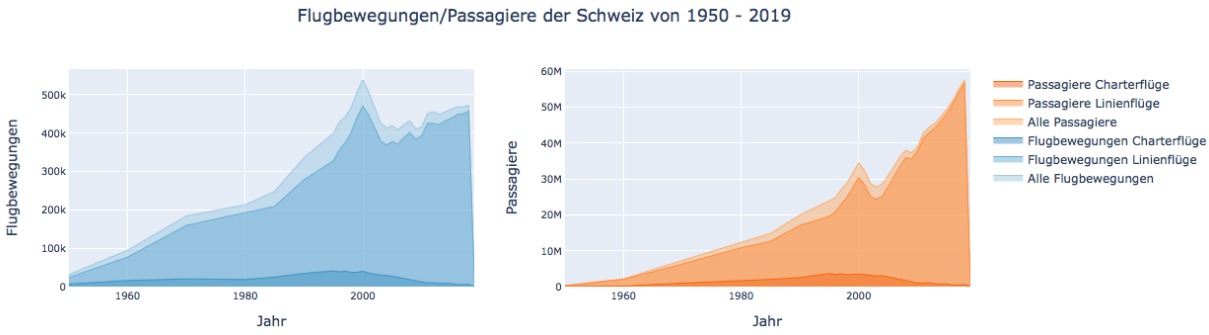
Die Visualisierung ist generell in 3 Hauptbereiche aufgeteilt. Die Bereiche sind wie folgt.

1. Flugbewegungen/Passagiere von 1950 – 2019 in je einer Grafik gruppiert nach verschiedenen Kriterien.
2. Flugbewegungen/Passagiere von 1950 – 2019 in einer Grafik gegenübergestellt und animiert.
3. Verschiedene Grafiken der Flugbewegungen/Passagiere pro Jahr dynamisch über Dropdowns auswählbar.

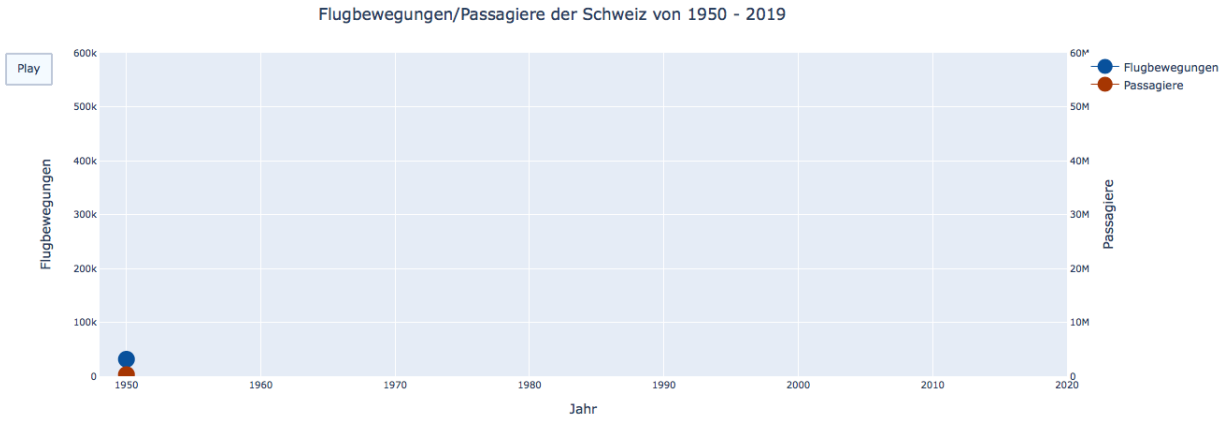
1. Bereich der Visualisierung

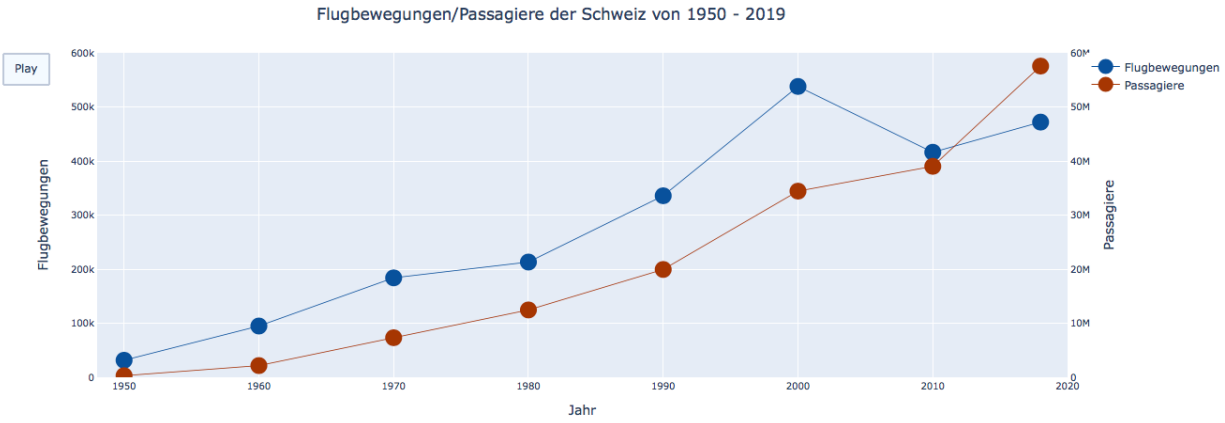
Der erste Bereich der Visualisierung zeigt alle Flugbewegungen und Passagiere von 1950 – 2019. Die Daten wurden nach folgenden Kriterien gruppiert:

- Flugbewegungen:
  - Alle Flüge
  - Linienflüge
  - Charterflüge
- Passagiere:
  - Alle Passagiere
  - Passagiere Linienflüge
  - Passagiere Charterflüge



Außerdem wird die Entwicklung der Anzahl Flugbewegungen und Passagiere in 10 Jahres Schritten in einer Grafik angezeigt. Die Animation kann mittels eines Start Knopfes gestartet werden.





2. Bereich der Visualisierung

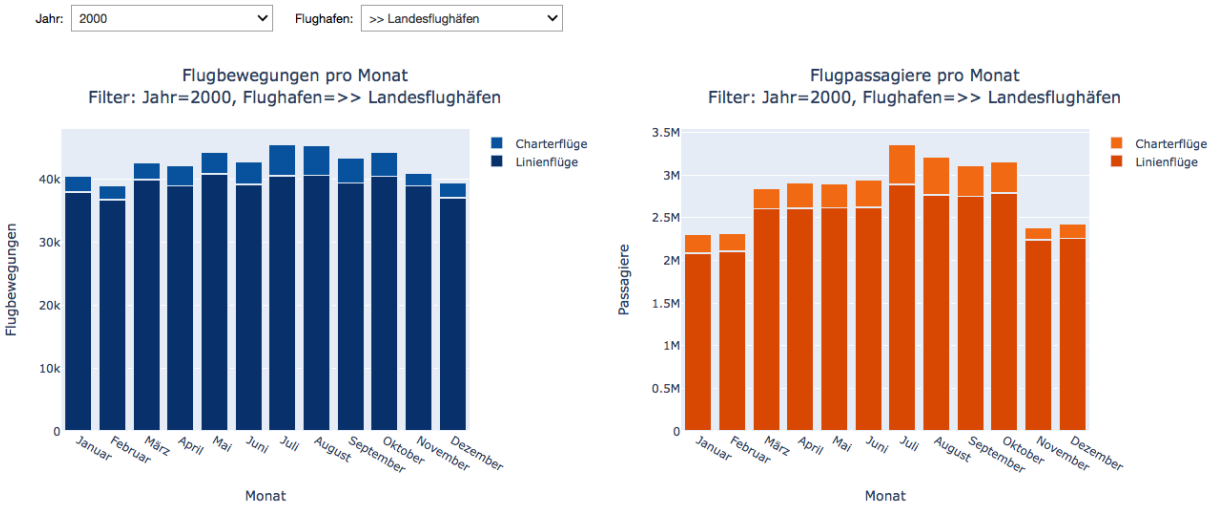
Dieser Bereich der Visualisierung konzentriert sich auf die einzelnen Jahre, selektierbar über Dropdowns. Das Hauptaugenmerk liegt hier einerseits in der Aufteilung nach Landesflughäfen und Regionalflugplätze sowie Linienflüge und Charterflüge sowie auch der Saisonalen Auslastung.

Flugbewegungen & Passagiere pro Jahr nach Flughafenkategorie

Diese Grafiken zeigen einerseits alle Flugbewegungen/Passagiere pro Jahr und Flughafenkategorie, Gruppiert nach Linienflüge und Charterflüge. Andererseits die Linienflüge und Charterflüge aufgeteilt nach Monat.

Jahresverlauf der Flugbewegungen & Passagiere nach Flughafenkategorie

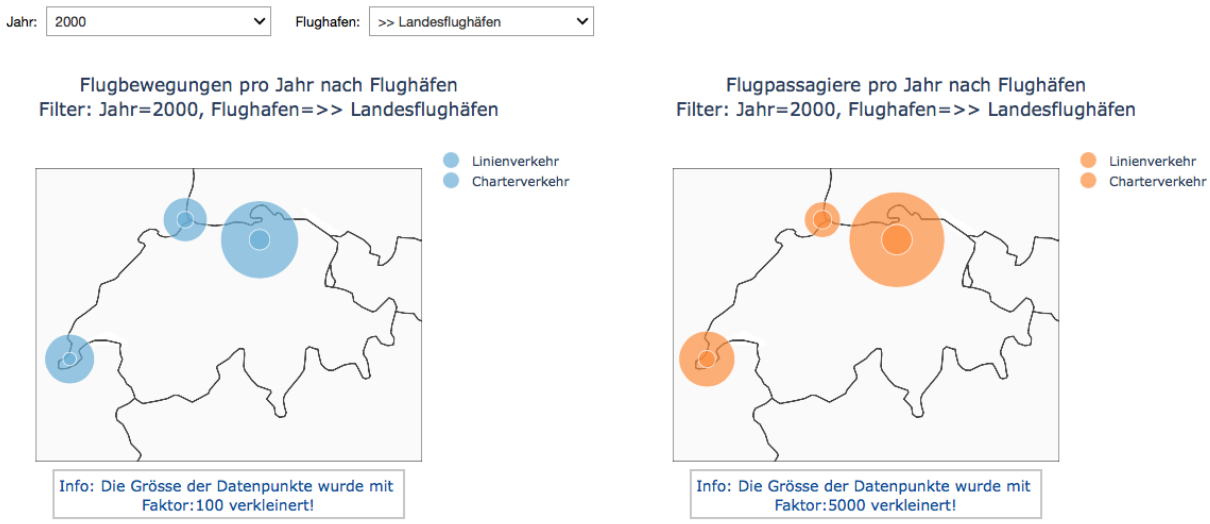
Der Jahresverlauf zeigt deutlich den Saisonalen Effekt der Linien oder Charterflüge. Während den Sommerferien sind die Flugbewegungen & Passagierzahlen höher als sonst.



3. Bereich der Visualisierung

Flugbewegungen & Passagiere pro Jahr nach Flughafen-Standort

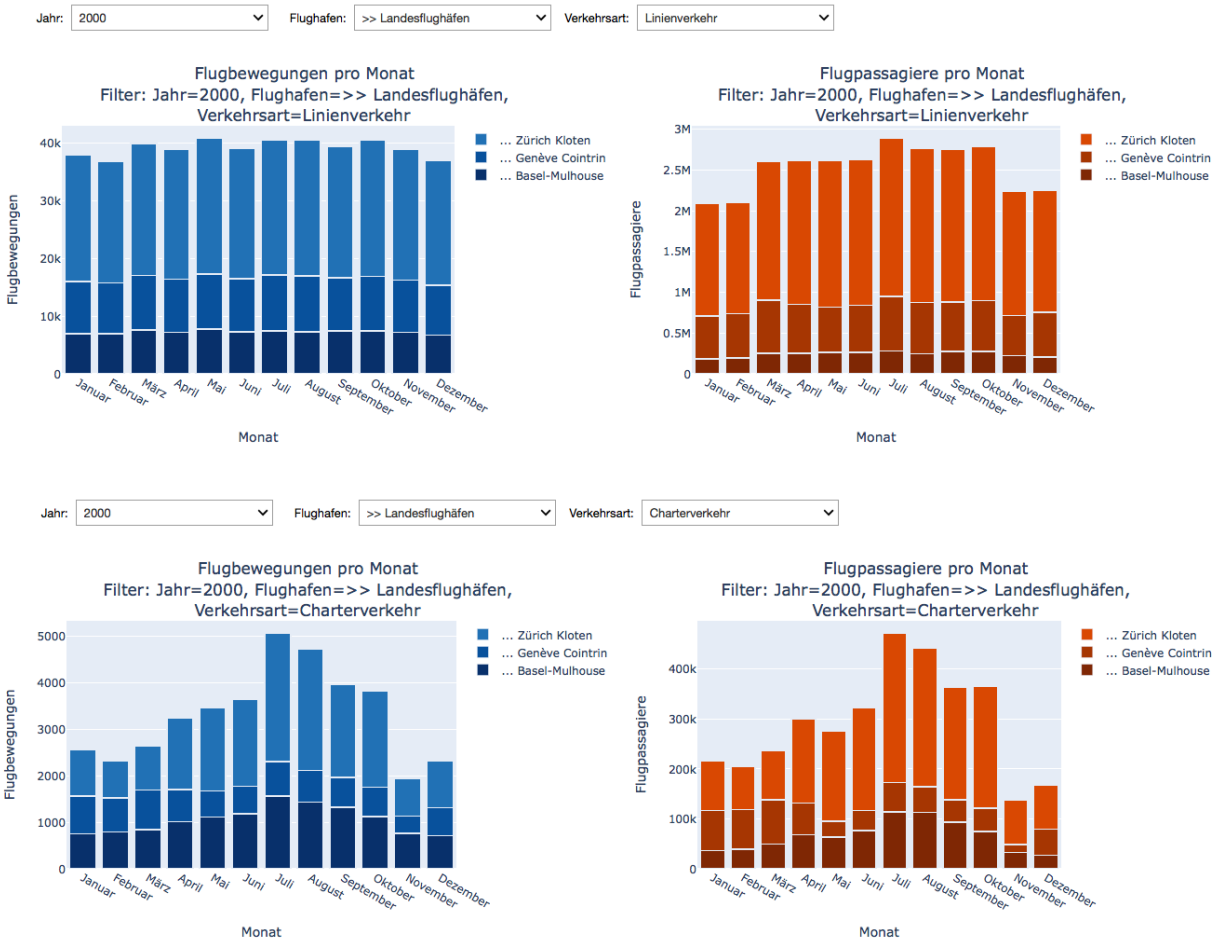
Hier werden die einzelnen Standorte mit der Größen Verteilung in je einer Scatter-Geo Grafik über das gesamte Jahr visualisiert.



4. Bereich der Visualisierung

Jahresverlauf der Flugbewegungen & Passagiere nach Flughafen-Standort

Das Schwergewicht dieser Grafiken liegt hier auf dem Saisonalen Effekt. Zum Beispiel wie verhalten sich die Flugbewegungen und Passagierzahlen während den öffentlichen Schulferien bei den einzelnen Flughäfen. Die Daten können jeweils für ein bestimmtes Jahr, eine Flughafenkategorie oder einer Verkehrsart gefiltert werden.



Code Struktur & Syntax

Die ganze WEB-Basierte Visualisierung basiert auf Python. Die Grafiken wurden alle mit Plotly, und die Animation mit Figure-Widgets realisiert. Generell kann der Aufbau in 5 Gruppen aufgeteilt werden.

- Importieren der Daten
- Aufbereiten der Daten
- Filtern der Tabellen
- Erstellen der Grafiken
- Aktualisieren der Grafiken

Importieren der Daten

Die einzelnen Tabellen werden über die Funktion «**imp_file**» einzeln importiert und in die entsprechenden DataFrames gespeichert. Siehe «**Code-Block-1**» im Jupyter Notebook.

Flugbewegungen:					
Flughafen:	**Art des Verkehrs:**	**Monat:**	**2000**	**2001**	**2002**
Schweiz	Linienverkehr	Januar	40115	41713	35536

Flugbewegungen:					
Flughafen:	**Art des Verkehrs:**	**Monat:**	**2000**	**2001**	**2002**
Schweiz	Linienverkehr	Januar	40115	41713	35536

Geo-Daten:		
Flughafen:	**Latitude:**	**Longitude:**
... Zürich Kloten	47.458216	8.555476

Aufbereiten der Daten

Nach dem Import der Daten in ein Pandas Dataframe müssen die Jahre in eine Kolonne „Jahr“ um konvertiert werden. Dies kann mit der Funktion **Melt** sehr einfach erledigt werden. Gleichzeitig wird die Kolonne auf Nummerisch umgestellt.

Flugbewegungen:				
Flughafen:	**Art des Verkehrs:**	**Monat:**	**Jahr:**	**Flugbewegungen:**
Schweiz	Linienverkehr	Januar	2000	40115
Schweiz	Linienverkehr	Januar	2001	41713
Schweiz	Linienverkehr	Januar	2002	35536

Flugpassagiere:					
Flughafen:	**Art des Verkehrs:**	**Art der Passagiere**	**Monat:**	**Jahr:**	**Flugpassagiere:**
Schweiz	Linienverkehr	Transitpassagiere	Januar	2000	45707
Schweiz	Linienverkehr	Transitpassagiere	Januar	2001	30646
Schweiz	Linienverkehr	Transitpassagiere	Januar	2002	28582

Filtern der Tabellen

Für die Filterung der Daten werden zuerst diverse Globale Variablen erstellt wie z. B. Definition aller benötigten Farben oder der Variable-Namen. Siehe «**Code-Block-2**» im Jupyter Notebook.

Danach werden die einzelnen Tabellen über die Funktion «**fltr_df**» nach den einzelnen Filtern abgefüllt. Siehe «**Code-Block-3, 8, 12, 15, 16**» im Jupyter Notebook.

Erstellen der Grafiken

Für den ersten Teil der Visualisierung wird keine Animation benötigt, daher müssen nur die Scatter-Charts erstellt und mit den entsprechenden Daten abgefüllt werden. Zuerst werden alle Variablen in Arrays gespeichert und danach die Traces für den Scatter-Plot in einer Schlaufe abgefüllt. Anschliessend wird das Layout definiert und der Figur zugewiesen. Siehe «**Code-Block-4, 10, 14, 17**» im Jupyter Notebook.

Der 2. Bereich der Visualisierung benötigt ein bisschen mehr Aufwand der Datenvorbereitung. Da hier für das Anzeigen der Daten der Filter jeweils anders definiert werden muss, habe ich diesen Teil über ein Mehrdimensionales Array definiert. Somit ist das Programm in der Lage die Traces dynamisch abzufüllen. Nach dem Anklicken des «Play» Knopfes werden jeweils die Frames abgefüllt und die Grafik aktualisiert. Siehe «**Code-Block-5, 10, 14, 17**» im Jupyter Notebook.

Dropdowns für die Filterung der Daten

Für die Dropdowns habe ich die Funktion «create_dropdown» erstellt, welches dann in allen Bereichen aufgerufen werden kann. Siehe «**Code-Block-6**» im Jupyter Notebook.

Die Dropdowns werden im nächsten Schritt erstellt und aus den entsprechenden DataFrames abgefüllt. Siehe «**Code-Block-7**» im Jupyter Notebook.

Aktualisieren der Daten & Grafiken

Anschliessend braucht es noch die Funktionen für das Aktualisieren der Daten und der Grafik. Wenn der Benutzer ein anderes Jahr im Dropdown auswählt, wird die «**Batch_update()**» Funktion von Plotly ausgeführt und die Grafik aktualisiert. Siehe «**Code-Block-10, 14, 17**» im Jupyter Notebook.

Für die Darstellung der Grafiken und Dropdowns wird die Figure-Widgets Library verwendet. Man Kann die Objekte beliebig horizontal oder vertikal zusammensetzen. Siehe «**Code-Block-10, 11, 14, 17**» im Jupyter Notebook.

Anpassen der Zellenbreite im Jupyter Notebook

Es gibt zwei verschiedene Möglichkeiten die Breite anzupassen. Die Erste Variante funktioniert nur für das gerade geöffnete Notebook. Die zweite Variante stellt die Breite bei allen Notebooks um.

Variante1:

1. Geben Sie folgenden Befehl in einer Zelle ein:

```
#*********************************************
from IPython.core.display import display, HTML
display(HTML("<style>.container { width:99% !important; }</style>"))
#*********************************************
```

Die Zellenbreite sollte danach automatisch auf 99% der Bildschirmbreite gestellt sein.

Variante2:

1. Gehen Sie zum Installationsverzeichnis des jeweiligen Benutzers z.B /home/benutzer/.jupyter
2. Erstellen Sie einen Ordner mit dem Namen 'custom'.
3. Mit einem Texteditor kann nun die folgende css Datei erstellt werden.

```
.container {
    width: 99% !important;
}

div.cell.selected {
    border-left-width: 1px !important;
}

div.output_scroll {
    resize: vertical !important;
}
```

4. Speichern Sie die Datei unter dem Namen 'custom.css' im custom Ordner.
5. Zum Schluss muss noch Jupyter neu gestartet werden.

Deployen der Webanwendung auf Heroku

Vorbereiten der Daten auf GitHub:

1. Einloggen auf GitHub: https://github.com/
2. Heraufladen der Dateien über den Button 'Upload Files'
3. Der Download Pfad zu den Dateien kann wie folgt herausgefunden werden.
    a. Auf eine Datei klicken z. B. Flughafen-Schzeiz.csv
    b. Auf Raw klicken und den Pfad kopieren: https://raw.githubusercontent.com/josefgulyas/ffhs/master/Flughafen-Schweiz.csv
4. In der Jupyter Notebook Datei den Pfad für die Imports anpassen.

Deployen der Jupyter Notebook Datei auf Heroku:

1. Jupyter notebook file nach '/home/josef/voila-heroku/notebooks' kopieren.
2. Bei Heroku einloggen: /snap/bin/heroku login
3. Im text file die Module, welche im Notebook benutzt werden, angeben.
    a. z.B. ipywidgets
    b. plotly
    c. Pandas
4. Das Text file befindet sich unter: '/home/josef/voila-heroku/requirements.txt'
5. Im Procfile muss noch der Name des zu kompilierenden Notebooks eingetragen werden.

```
z.B. : web: voila --port=$PORT --no-browser --enable_nbextensions=True
notebooks/DaVi_prototype_Josef_Gulyas.ipynb
```

Das Procfile befindet sich unter: '/home/josef/voila-heroku/Procfile'

6. Das lokale Verzeichnis als Repository initialisieren mit: git init
7. Ein neues Repository auf Heroku erstellen: /snap/bin/heroku create "davi-semesterarbeit-jgulyas"

8. Git mit dem neu erstellten Repo verknüpfen: /snap/bin/heroku git:remote -a "davi-semesterarbeit-jgulyas"
9. Das Notebook mit GitHub hochladen: git commit -am "davi-semesterarbeit-jgulyas"
10. Das Notebook deployen: git push heroku master

Reflexion des eigenen Vorgehens

Ich habe mir schon bei der Auswahl des Themas zur Visualisierung sehr viel Zeit genommen. Es war gar nicht so einfach etwas zu finden bei welchem ich verschiedene Arten von Grafiken anwenden kann. Die Vielzahl an Themen ist schier unendlich und trotzdem wollte ich etwas aussuchen, was mir Spass macht und ich auch was davon verstehe. Nach etlichem Stöbern habe ich mich dann für die Visualisierung des **Schweizerischen Luftverkehrs von 1950 – 2019** entschieden.

Da ich eventuelle Diskussionen über Data Governance vermeiden wollte, habe ich mich dann an den Daten vom BFS orientiert. Die Vielfalt von Themen auf der Homepage des Bundesamts für Statistik ist enorm und die Möglichkeiten der Datenkonfiguration und des Downloads finde ich auch sehr gut.

Die Entscheidung mit welchem Tool ich die Visualisierung realisieren wollte, war für mich sehr schnell entschieden. Die Rahmenbedingungen waren wie folgt:

- **Es musste ein Tool sein, welches ich noch nicht im Detail kenne.**
- **Das Tool sollte auf einem Mac oder Linux laufen.**

Da bei den meisten Tools die zweite Bedingung nicht erfüllt war, habe ich mich dann für Python entschieden.

Die Einarbeitung war nicht immer ganz einfach. Zum Glück existieren genügend Beispiele auf der Plotly Homepage, wo man die Vorgehensweise und die Syntax studieren kann. Die grösste Schwierigkeit fand ich herauszufinden, welche Möglichkeiten man mit dem Tool hat und wie man es dann auch umsetzt. Ich musste mehrere male die Struktur des Codes umstellen, da es wie Sie erwähnt haben, zu stark an copy & paste ausgesehen hat. Ausserdem erhöht es die Fehleranfelligkeit. Die Aktualisierung der Grafiken erscheint mir Problematisch. Ich hatte oft Probleme mit der Performance. Man darf nicht zu viele Grafiken auf einmal über ein Dropdown aktualisieren. Dies führt schnell mal zu grossen Wartezeiten. Ein grosses Problem finde ich noch, dass die FigureWidgets nach dem nbconvert in ein html nicht mehr funktionieren. Ich hoffe dieses Problem wird mal in Zukunft ausgebessert.

Fazit:

Python ist eine sehr mächtige Programmiersprache mit unzähligen Erweiterungen und Möglichkeiten. Es war die richtige Entscheidung meine Visualisierung mit Python zu Realisieren. Ich habe sehr viel gelernt und es hat Spass gemacht mal was Neues kennenzulernen.