



# Certification CDSD

April 1<sup>st</sup>, 2025 – Louis Le Pogam





# Agenda

- **Block 1 - Build & Manage a Data Infrastructure – Kayak Project**
- **Block 2 – Exploratory Data Analysis – Steam Project**
- **Block 3 – Supervised Machine Learning – Conversion Project**
- **Block 3 – Unsupervised Machine Learning – Uber Pickups Project**
- **Block 4 – Deep Learning – AT&T Spam Detection Project**
- **Block 5 – Deployment – Getaround Project**
- **Block 6 – Lead a Data Project – Offensive Speech Recognition**



# Project Reminder

## Project

- Kayak Marketing Team would like to create a holiday recommendation application based on :
  - Weather
  - Hotels in the area
  - Based on real-time data

## Goal

- The data are not available and the goal is to get the needed data as following:
  - Scrape data from destinations.
  - Get weather data from each destination.
  - Get hotels' info about each destination.
  - Store all the information above in a data lake.
  - Extract, transform and load cleaned data from your datalake to a data warehouse.



# 4 building blocks for the data scrapping model

## Description

---

1

### Geolocalisation

- Processing of list of cities by obtaining GPS coordinates and INSEE codes with API
- Saving all data to CSV for weather queries

2

### Weather Data

- Retrieving 7-day weather forecasts for cities based on INSEE codes
- Ranking cities based on customizable criteria and creates aggregated rankings
- Saving ranked list based on number of favorable days to CSV

3

### Booking Website Scrapping

- Taking the top 5 cities based on previous analysis
- Searching for hotels in each city available on booking.com
- Saving in into a JSON file

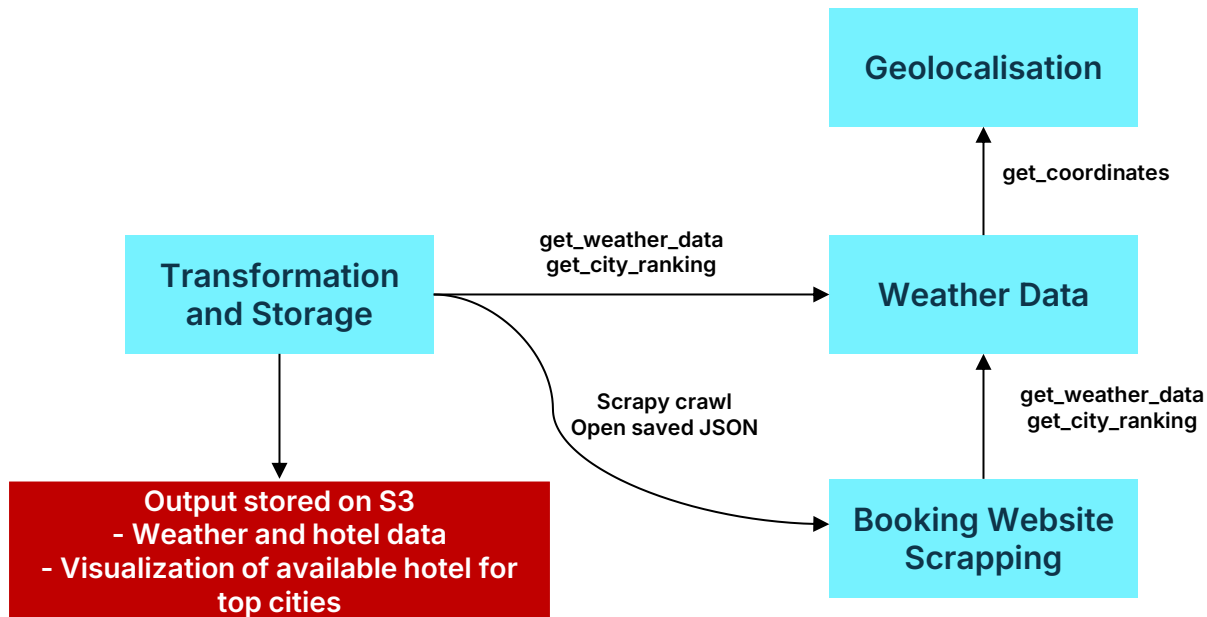
4

### Transformation and Storage

- Processing previous data by removing low-quality hotels and identifying consecutive night availability
- Creating interactive visualizations and uploads all results to AWS S3 storage.



# All 4 blocks are used to get the final output





# The weather module scraps the weather by city and ranks them based on defined ideal conditions

## get\_weather\_data

---

- Reads a list of French cities from a file based on the INSEE code
- Connects to a weather service (Meteo Concept API) for each city
- Gathers 7-day forecasts including:
  - Daily rainfall predictions
  - Temperature highs and lows
  - Wind speed information
- Organizes everything in a DataFrame where each row represents one day's weather forecast for a specific city

**Output : DataFrame with weather prediction for eachs targeted cities**

## get\_city\_ranking

---

- Flags Less-Than-Ideal Weather Days when:
  - Too hot / Too cold / Too rainy / Too windy
  - Default value: 35°C / 20°C/ 10mm / 50 km/h
  - Can be changed depending on the season
- Calculates Problem Days for each city over the forecast period
- Ranks Cities by:
  - Fewest problematic weather days
  - Lowest average rainfall as tiebreaker

**Output : DataFrame with a prioritized list of cities with optimal weather conditions for travelers**



# 700 pages scrapped to get the available hotels of the next 7 days

## Get Weather Data

- Open file of ranked cities and select top 5

## Get URL for each city and date

- Loop on each cities and for the next 7 days
- Get the URL of the search for each city x date

## Get the Details from the Search Results

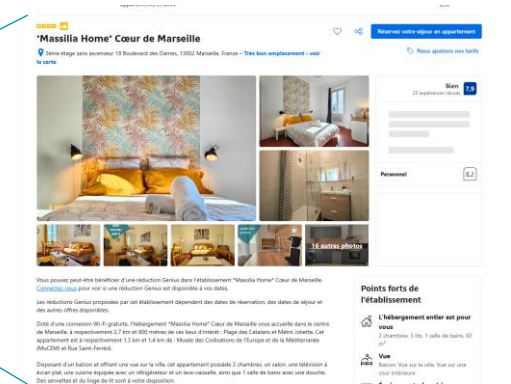
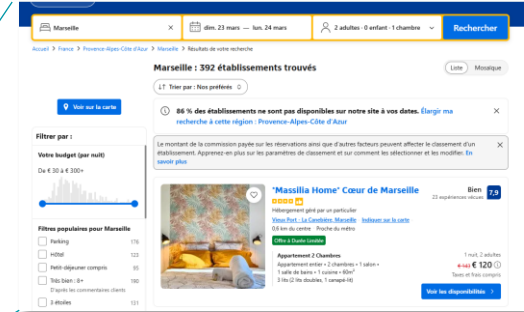
- For each results on the search page, get the main info : Name, ranking, price, distance...
- Keep only 20 first results to limit output size

## Open each page and get detailed data

- For each hotel, open the hotel page to get detailed info : Address, latitude, longitude, description, URL...

## Store Data in a JSON file

- Gather all data in one JSON
- 700 pages scrapped (5 cities x 20 hotels x dates x 7 dates) in 6 minutes





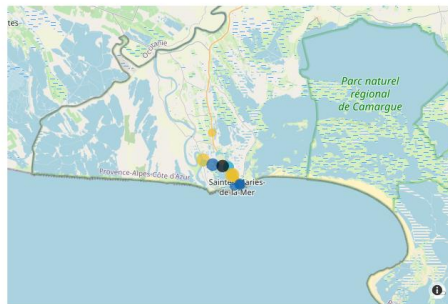
# All outputs are stored on a AWS S3 and ready to be used

## Output 1 : All database

- Dataframe with weather forecast
- Dataframe with City Ranking
- JSON with all booking.com data scrapped

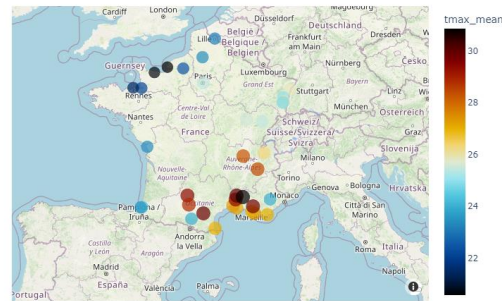
## Output 2 : Visualization of available hotels for each top 5 cities

Hotels available in Saintes Maries de la mer



## Output 3 : Overview of average temperature for the next 7 days of all targeted cities

Average max temperature for the next 7 days per city



All outputs are stored in a S3 and ready to be extracted and used





# Agenda

- Block 1 - Build & Manage a Data Infrastructure – Kayak Project
- Block 2 – Exploratory Data Analysis – Steam Project
- Block 3 – Supervised Machine Learning – Conversion Project
- Block 3 – Unsupervised Machine Learning – Uber Pickups Project
- Block 4 – Deep Learning – AT&T Spam Detection Project
- Block 5 – Deployment – Getaround Project
- Block 6 – Lead a Data Project – Offensive Speech Recognition



# Project Reminder



## Project

- Ubisoft is requesting an analysis on the games available on Steam to create the perfect game

## Goal

- The target is to identify and analyze the key factors influencing video game popularity and sales through a multi-dimensional market analysis examining:
  - Publisher dominance and release strategies
  - Game ratings and critical reception patterns
  - Temporal release trends, including pandemic impacts
  - Pricing strategies and discount effectiveness
  - Language availability and regional targeting
  - Age ratings and content restrictions
  - Genre popularity, performance, and publisher preferences
  - Platform distribution and genre-platform relationships



# Key Insights



## Steam sales macro- overview

There are around 55k games available on Steam with strong growth between 2012 and 2020 but limited ever since, especially following covid

- The CAGR of the number of games was 77.2% p.a. between 2012 and 2017 and +4.4% p.a. between 2017 and 2022
- The growth were stopped in 2022 where there were a huge dropped in number of games released this year
- We can assume that this was following the decrease in development in the covid period in 2020-2021 that delayed games supposed to be launched in 2022

Steam has strongly contributed to market access of small developers

- The #1 genre is Indie game
- Large majority of developer and published have created between 1 and 3 games
- The growth in number of games has been driven by games with below 20k owners
- However, this seems to still be a very competitive market and there is a high risk of over-supply and bankruptcy among small developers

Most of the games have less than 20k owners

- All this small games have a very low number of rating, which would limit the reach to new customers
- In 2022, Indie Genre games have known one of the highest decrease in number of games released



# Key Insights



## Trends

The overall trend shows that small developers are shifting from free-to-play towards early access to ensure cash collection:

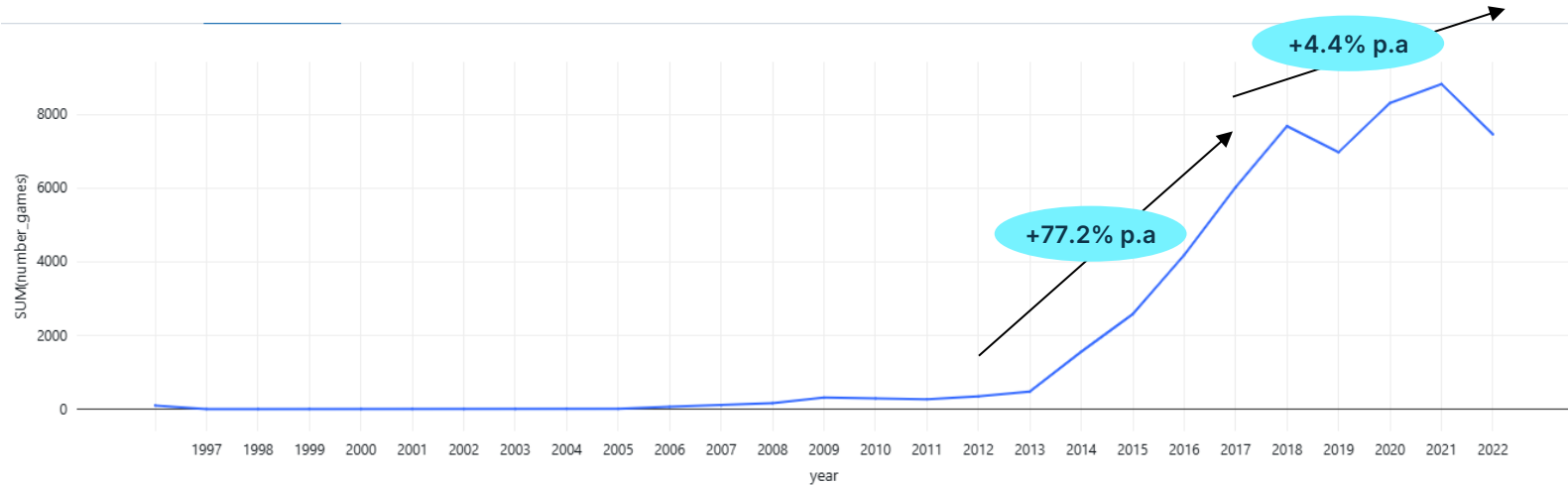
- There is a current over-supply in the gaming market has led to a reduction in game releases in 2022
- This has mostly impacted free-to-play games and games with low price tags and production values
- Early access allows developers to be closer to the community, better finance game development, and correct bugs with direct feedback



After a massive increase between 2012 of 2017, the number of new release is getting slower with a reduction post covid



## Number of new games in Steam | in number of games, 1997-2023

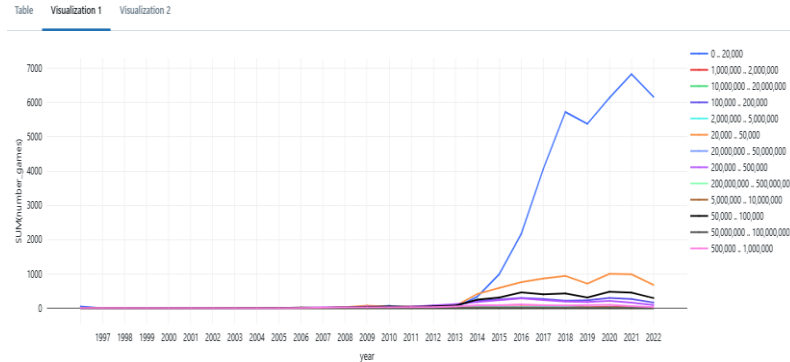




Steam has allowed all developer to release then game leading to a strong growth of small games and a majority of existing games owned by less than 100k players

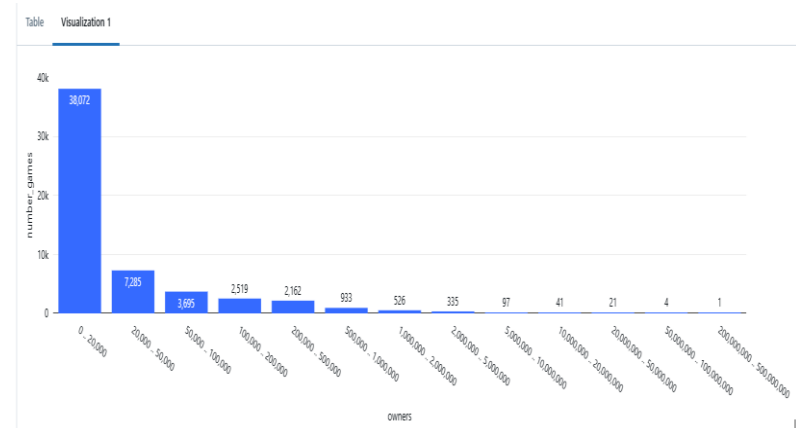


Number of new game in Steam per nbr of owners |  
in number of games, 1997-2023



Main growth of number of games coming from games owned by less than 20k users

Number of new game in Steam per nbr of owners |  
in number of games



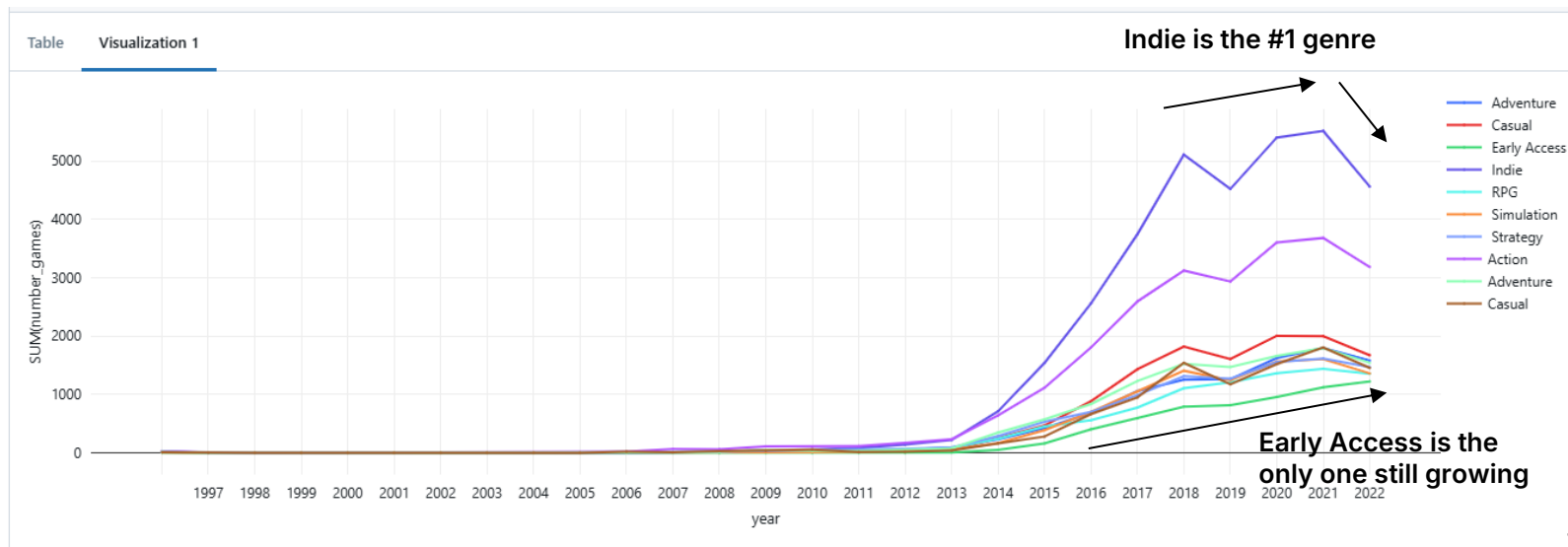
Majorities of game are owned by less than 100k users



Indies is the top 1 genre which has driven the growth but also shown a strong decrease in 2022; Early access still growing to insure game development



Number of genre tagged on each games in Steam |  
in number of tags, 1997-2023, one game can have several tag

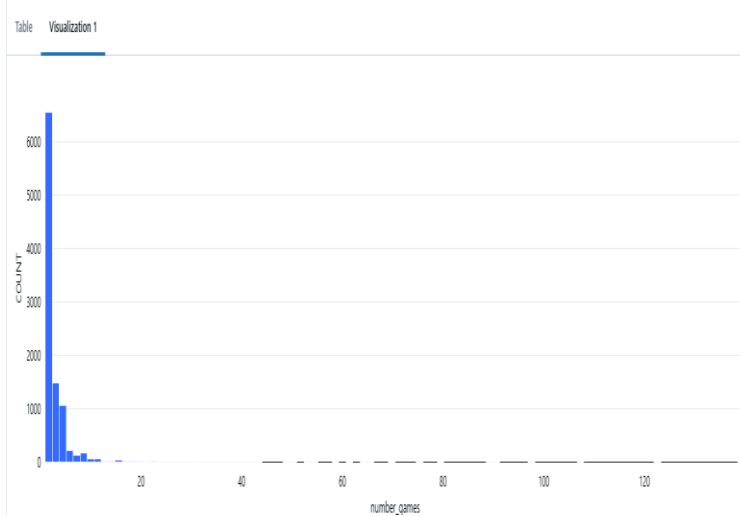




Large majority of developers have released just a couple of games in Steam with a low number of ratings

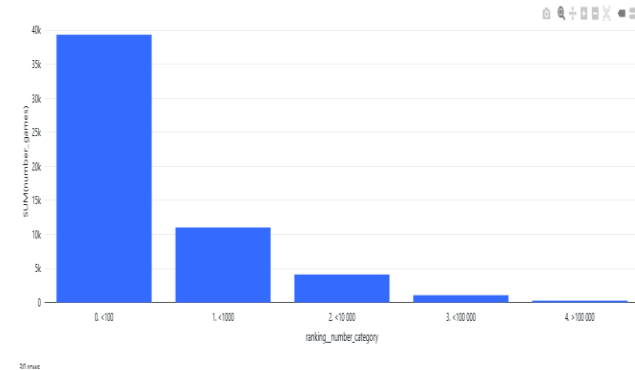


Histogram of number of games per developer | 1997-2023



*Main category between 1 and 3 games*

Histogram of number of games per number of rating | 1997-2023



*70% of games have less than 100 ratings*





# Key Insights



## Business Model

Overall, the market seems to be split into two main distribution modes with their own business models:

- Free-to-play games rely on the number of concurrent users to generate revenue from in-game sales
- Paid games are driven by game price tags, discounts, and number of owners
- The free-to-play market is highly competitive with a large number of games that do not seem profitable

The F2P market is highly concentrated, with 2 games gathering half of the users and 10 games covering 84% of the total users:

- More than 7.4k free-to-play games have fewer than 100 concurrent users, making it difficult to be profitable
- Only 100 games out of the 7.7k F2P games available have more than 1,000 concurrent users
- Free-to-play game releases strongly decreased between 2020 and 2022

For paid games, there are sub-segments depending on the price tag, which are also very competitive:

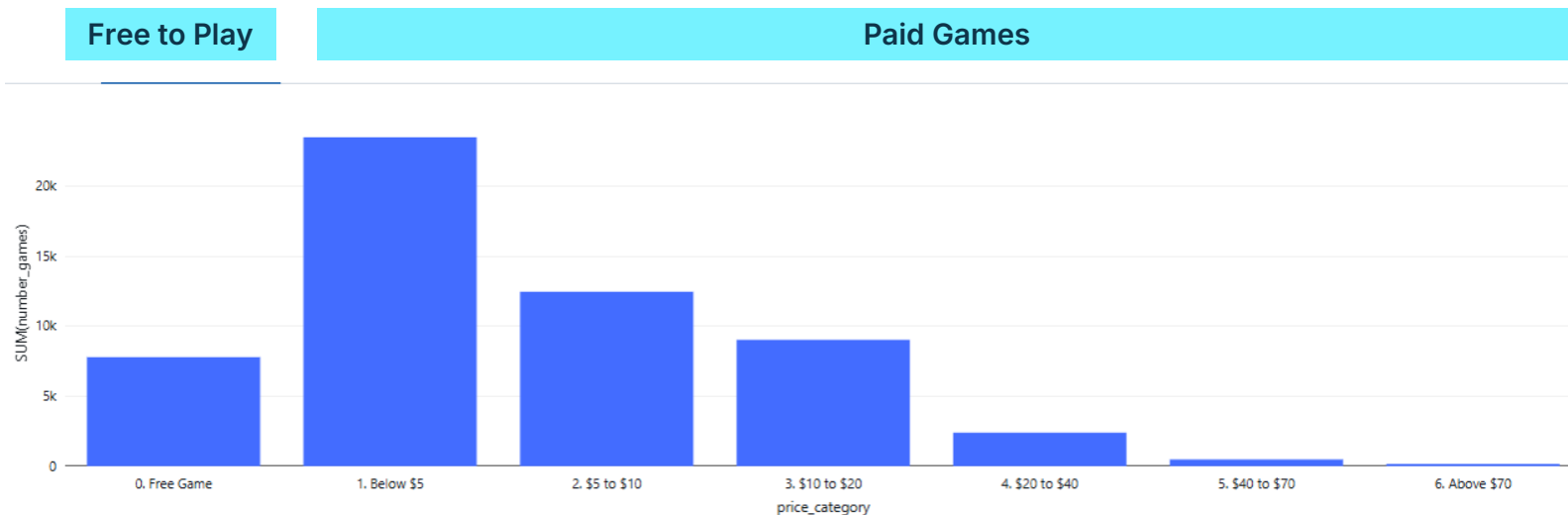
- Prices can vary from a few dollars to more than \$70 and reflect the production value: the more it costs, the better value for money the player expects
- Based on a hypothesis to be confirmed, there is a direct correlation between price tag and development costs: the higher the production value, the higher the cost
- For each segment, only 15% to 30% of the games appear to be profitable, showing how competitive the market is



## 2 different business models : Free to play games and paid games



Number games per price category|  
in number of games, 1997-2023,

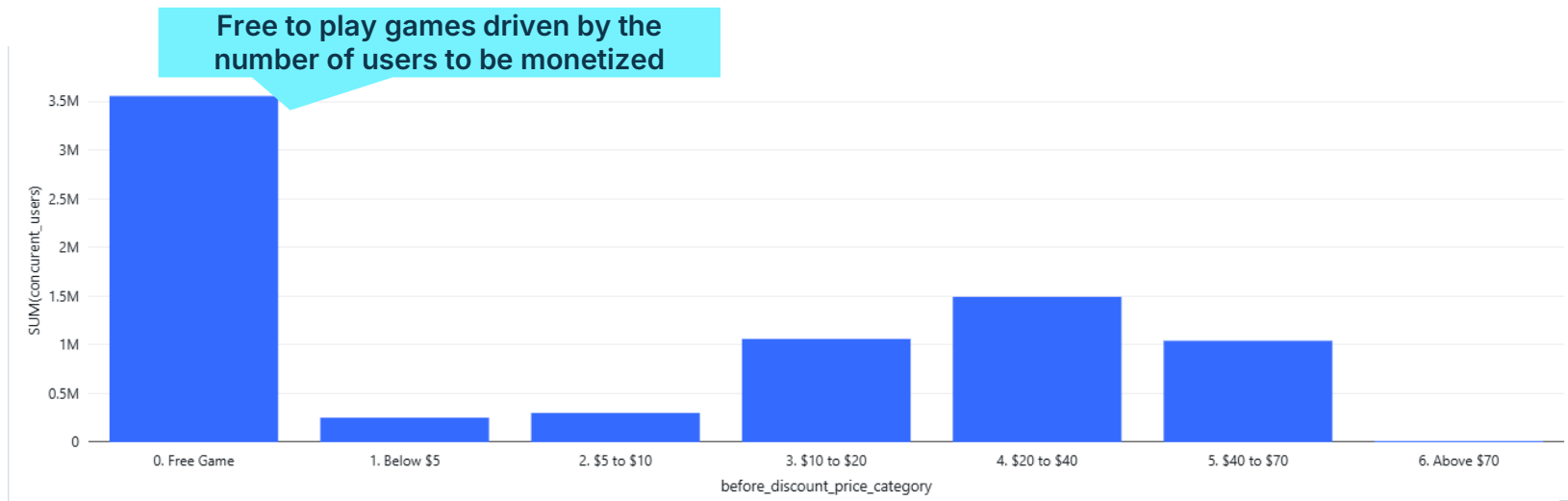




## Largest number of concurrent users are on Free to Play games, used for monetization



Number of concurrent users per price category |  
in number of games, 1997-2023,

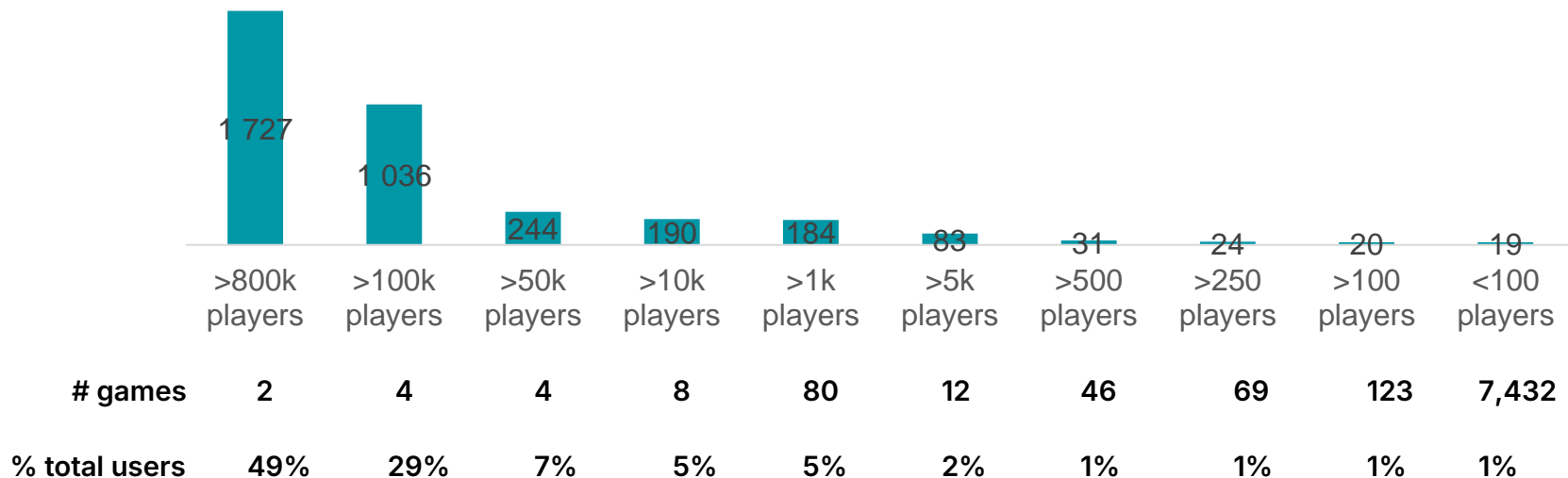




Free to play games is very concentrated as 2 games have 50 % of all users and 6 have 80% of the total



### Number of concurrent users per category of games |

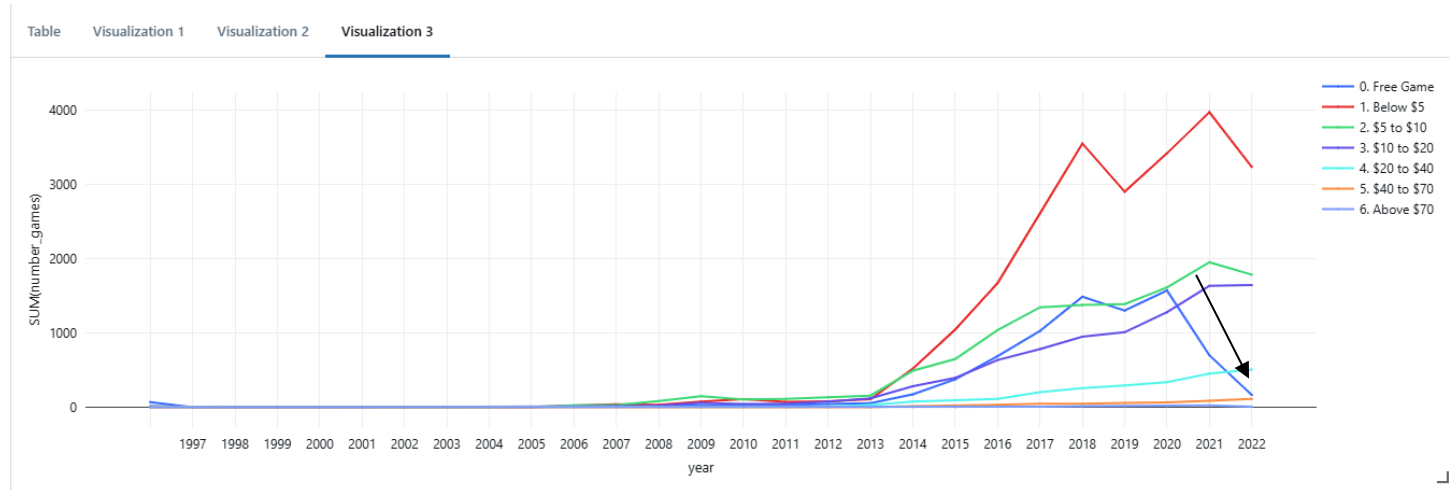




Free to Play release have strongly decrease showing the difficulty to handle this type of games



## Number of game release per year per price category |



Strongest decrease on the F2P segment

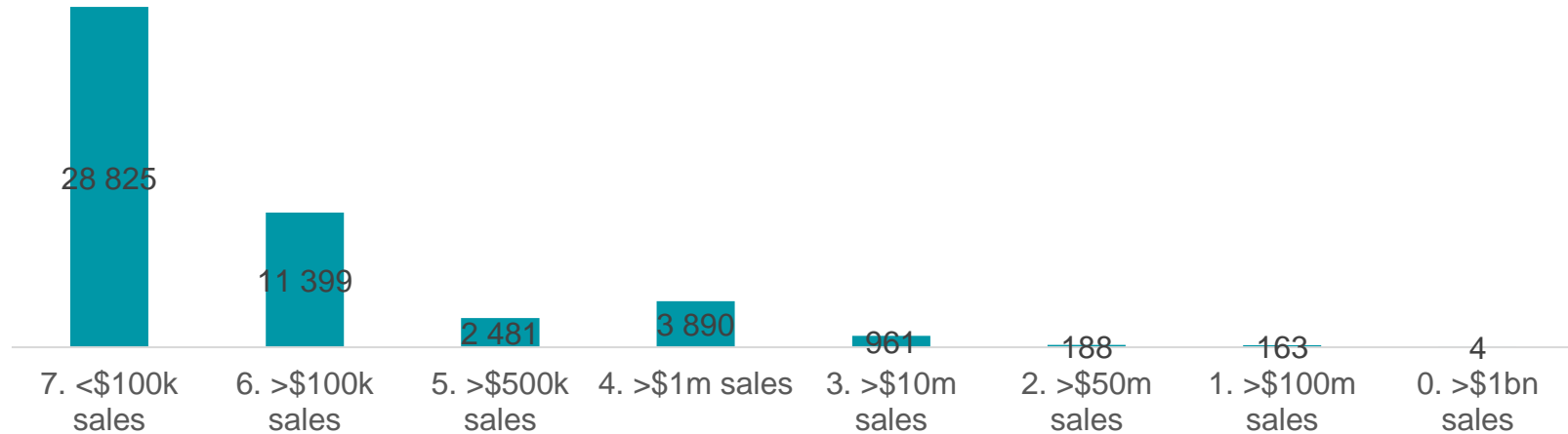


## Majority of the games are creating less than \$100k revenues



### Number of games users per category of games |

Sales estimation based on number of owner and price after discount



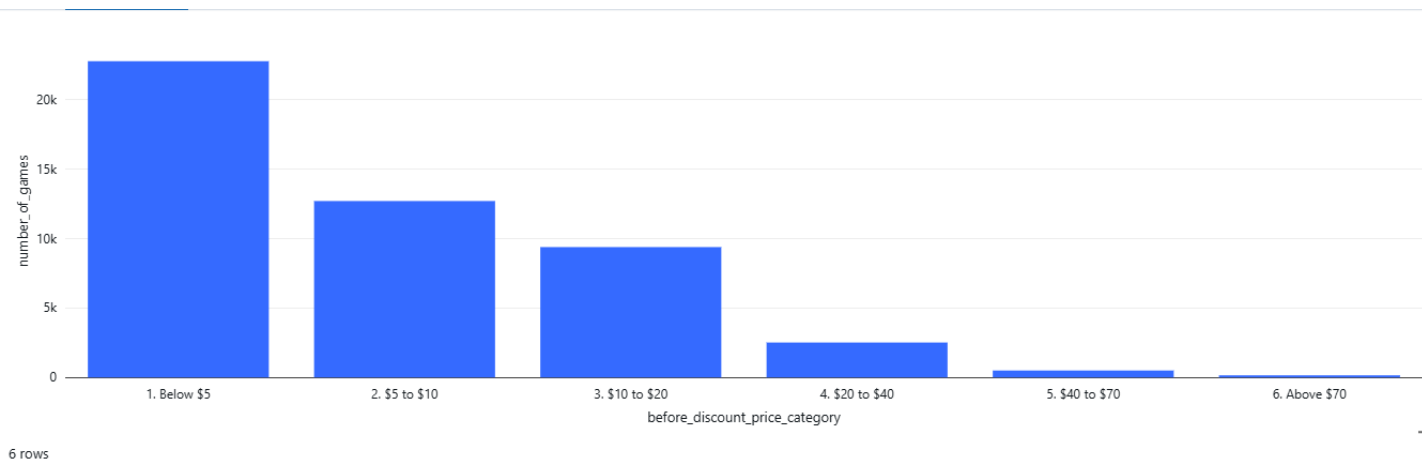


Majority of games are below \$5 with an average revenues below \$100k. High average turnover per game is between \$40 and \$70



### Number of games users per price after discount |

Sales estimation based on number of owner and price after discount



Average Revenues  
per game in \$m

0.1

0.7

3.0

12.1

39.5

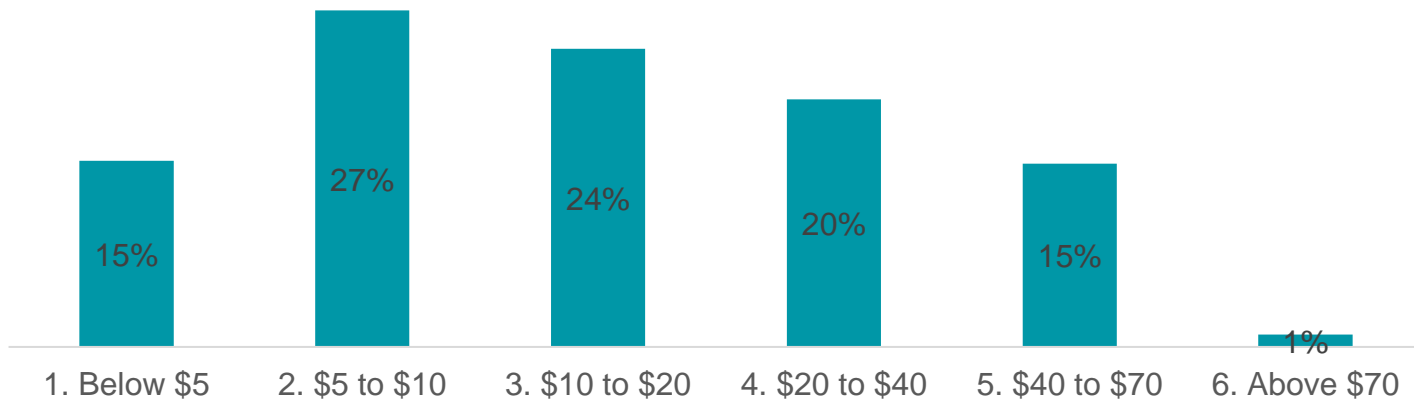
10.9



In average, less than 30% of games of each category are profitable



### Percentage of profitable project based on estimated cost | Preliminary profitability threshold



Profitability  
Threshold in \$m

0.1

0.1

1

10

50

70





# Key Insights



## Ubisoft Review

Given Ubisoft's partnership with Epic, the Steam sales dataset seems not relevant to analyze gaming market and Ubisoft's position

- Ubisoft had not game released on Steam since its partnership with Epic in 2020
- The market share of Ubisoft cannot be analyzed as there are almost 3 years' worth of release missing in the dataset
- Given Ubisoft decision, we can assume there are other case like this one and therefore, the dataset is not representative of the market

Based on the data before 2020, we can see that Ubisoft is the #1 published in revenues with \$3.6bn in total, with a strong focus on paid games related to big franchise

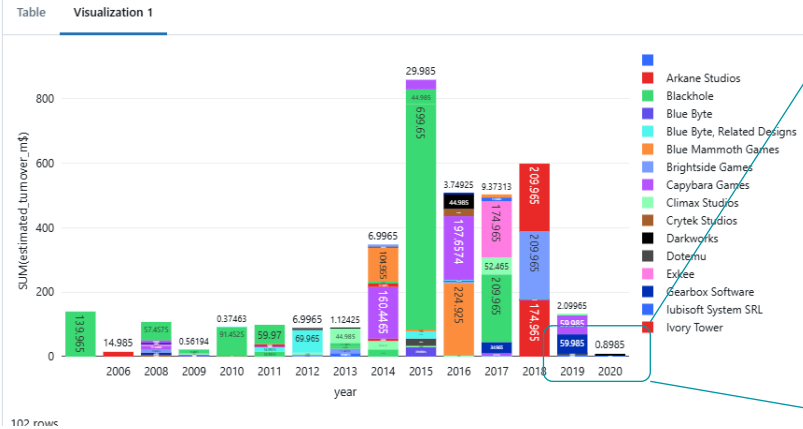
- Ubisoft's games have made above \$3.6bn between 2006 and 2020 with a peak in 2015, with the release of "Tom Clancy's Rainbow Six Siege"
- On the free to play games, Ubisoft has only one significant game with around 10k users, in 18th position of the F2P games
- 61% of sales are concentrated into 3 franchises: Tom Clancy's, Assassin's Creed and Far Cry
- However, there are no information of Ubisoft release since 2020



# No data from Ubisoft after 2020 as the publisher left Steam in 2019



Ubisoft estimated turnover by development company |  
in \$bn, based on price before discount and number of games owners, list of dev company not exhaustive



WELCOME BACK

## Ubisoft comes crawling back to Steam after years on Epic Games Store

In 2019, company said Valve's 30 percent commissions were "unrealistic."

KYLE ORLANDO - 22 NOV 2022 11:21 211

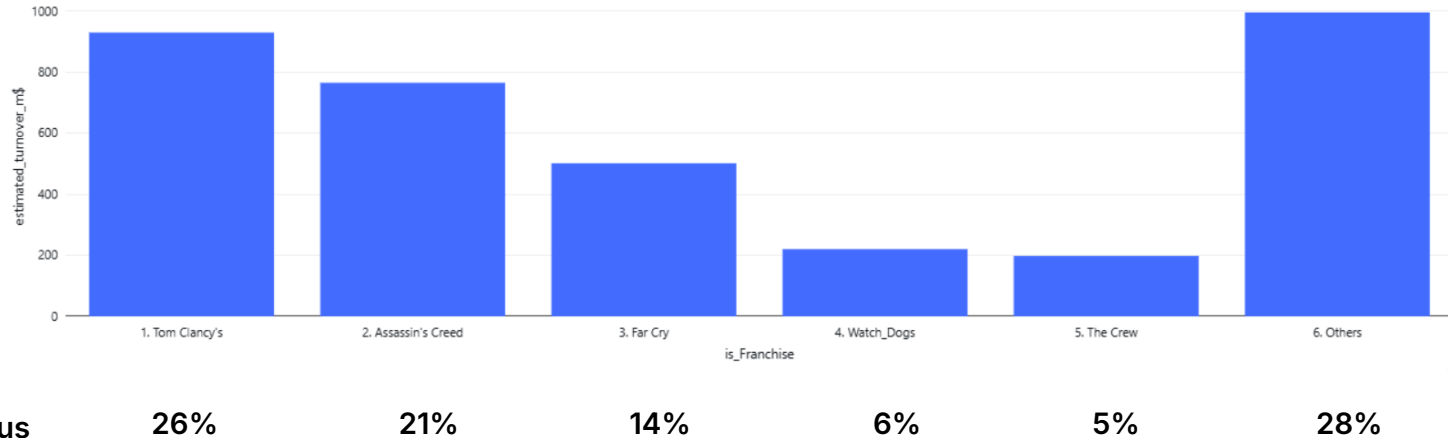




# Ubisoft paid games are focused on main franchise with 60% of revenues in 3 of them



Ubisoft estimated turnover by franchise |  
in \$m, based on price before discount and number of games owners

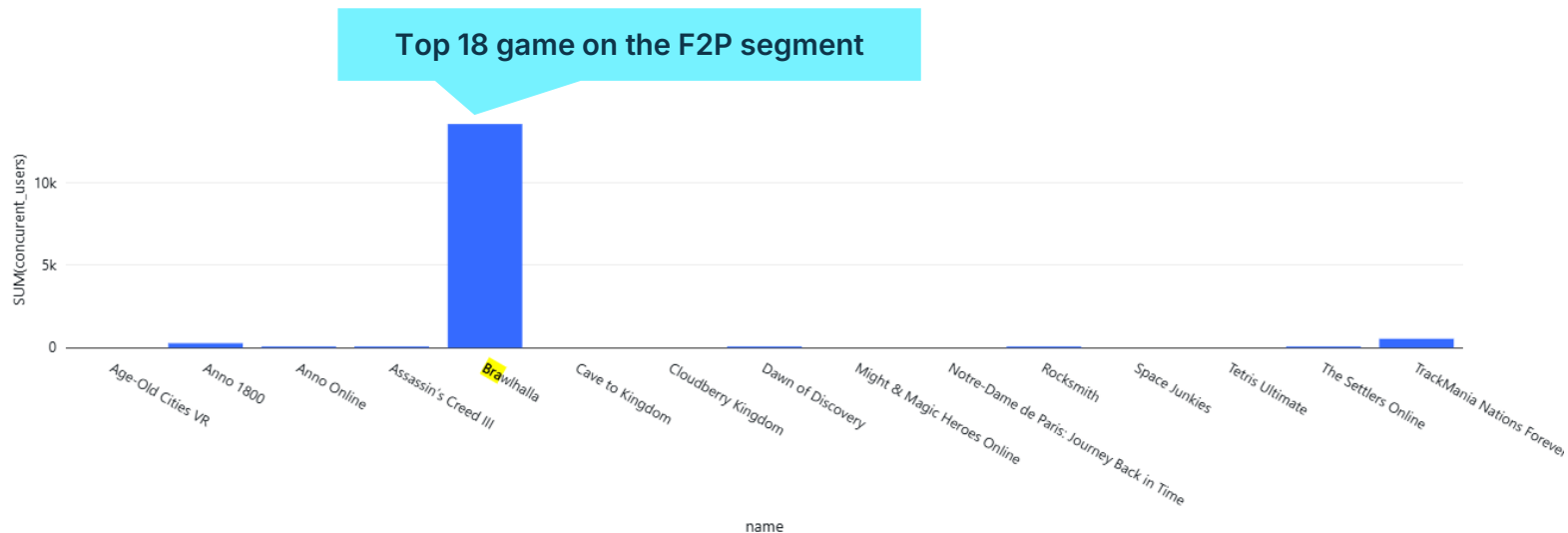




# Ubisoft has only 1 important F2P game, which is in 18<sup>th</sup> position in the market



Number of concurrent users on Ubisoft free to play games|





# Agenda

- Block 1 - Build & Manage a Data Infrastructure – Kayak Project
- Block 2 – Exploratory Data Analysis – Steam Project
- Block 3 – Supervised Machine Learning – Conversion Project
- Block 3 – Unsupervised Machine Learning – Uber Pickups Project
- Block 4 – Deep Learning – AT&T Spam Detection Project
- Block 5 – Deployment – Getaround Project
- Block 6 – Lead a Data Project – Offensive Speech Recognition



# Project Reminder

## Project

The newsletter team wants to understand user behavior on their website and build a predictive model for subscription conversions using minimal user information. Their goals are to:

- Identify key features that explain subscription behavior through model parameters analysis
- Discover potential new levers to improve conversion rates

They've created a competition where teams build predictive models using an open-sourced website traffic dataset. The competition uses the F1-score as the evaluation metric to balance precision and recall in predicting user subscriptions

## Goal

This project builds a machine learning model to predict customer conversion rates based on user behavior and demographic data. Here are the main steps

- Exploratory Data Analysis
- Data Preprocessing
- Model development and comparison
- Final model and predictions



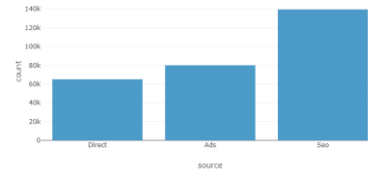
# First analysis of the dataset is showing non missing data and a low conversion rate

## Dataset description

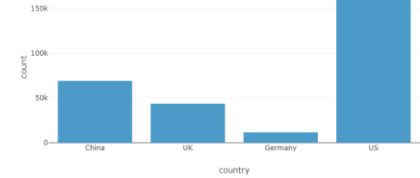
- There are 6 columns with 284k lines and no missing data :
  - Country : User origin among China, Germany, UK and US
  - Age : Age of the user
  - New\_User : If it is a new user or not
  - Source : How the user were obtained
  - Total\_pages\_visited : Nbr of pages visited
  - Converted : Target
- First insights
  - US and China are the main countries in term of visitors
  - SEO is the main source
  - The visitors are visiting just a few pages in average

## Distribution of main columns of the dataset

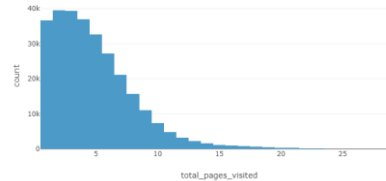
Source



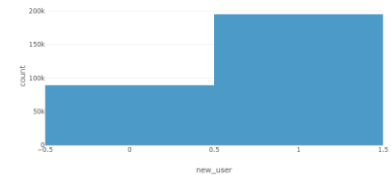
Country



Page Visited



New User

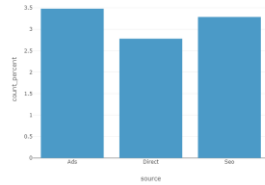




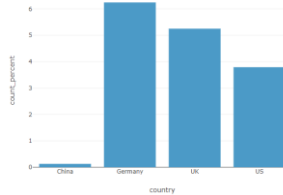
The conversion rate is the higher for GE and the UK, for historic customer and user visiting a large number of pages

## Conversion rate in % depending on the column

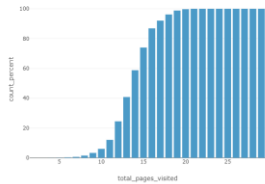
Source



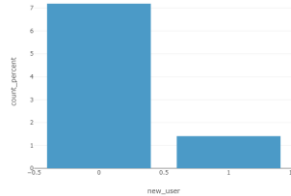
Country



Page Visited



New\_user



## Key Insights

- The highest converting rate is coming from Germany and the UK. There are almost no conversion in China
- Each source seems to have the same conversion rate
- High conversion for historic customer
- The highest the number of page, the higher the conversion rate. After 20 page visiting, there is a 100% conversion rate





# After testing different model, the best performance were achieved with Logistic Regression with a f1 of 0.76

## Mode selection

### Preprocessing

- Standard preprocessing :
  - Standard Scaler for numerical data
  - One Hot Encoder for categorical data
- Train and Test split

### Experiments

- Several model testes
  - Logistic Regression
  - Random Forest
  - Decision Tree
  - Ada Boost
  - XGBoost
- Cross validation performed on each model to get the best parameters
- Model selected based on f1 score

## Results

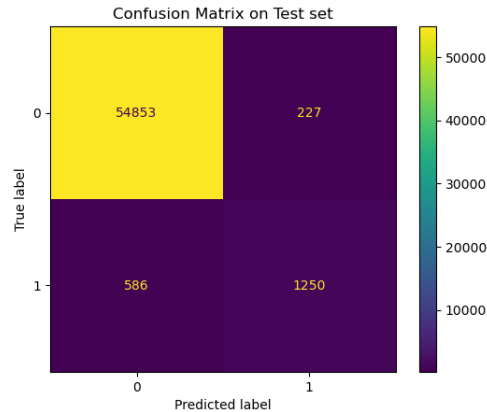
|   | Algorithm           | Accuracy_mean | F1_mean  | Recall_mean | Precision_mean |
|---|---------------------|---------------|----------|-------------|----------------|
| 0 | Logistic Regression | 0.986313      | 0.765072 | 0.690904    | 0.857096       |
| 1 | Random Forest       | 0.984275      | 0.731542 | 0.678377    | 0.804380       |
| 2 | Decision Tree       | 0.983651      | 0.718556 | 0.644744    | 0.810237       |
| 4 | Ada Boost           | 0.984069      | 0.699537 | 0.575572    | 0.892756       |
| 3 | XGBoost             | 0.979000      | 0.529437 | 0.368600    | 0.950936       |

**Best performance with the Logistic Regression  
with a f1 of 0.76**



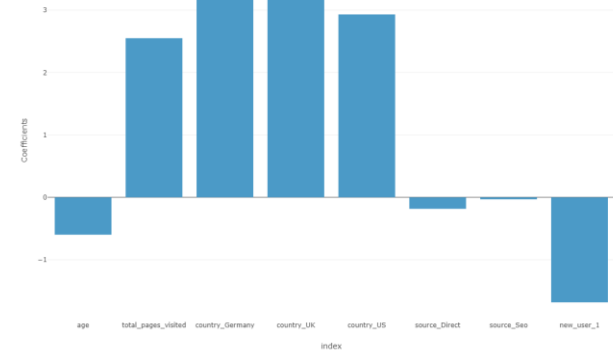
# Model confirmed EDA first conclusions

## Confusion Matrix



Lots of converted customers not correctly created leading to a recall of only 0.69

## Coefficient per column comparison



Model confirmed the importance of the country, the number of page and the type of user  
Age is slightly less important



# Business recommendation

## Improve Chinese Market Experience

- Investigate localization needs or technical improvements
- Address product-market fit issues

## Focus on User Retention

- Implement strategies to encourage repeat visits
- Develop loyalty programs or incentives

## Enhance User Engagement

- Redesign website to encourage deeper navigation
- Create compelling content paths across multiple pages

Business Recommendations (continued)

## Age-Targeted Marketing

- Consider campaigns specifically targeting younger demographics
- Adapt content style and offerings for different age groups

## Channel Optimization

- Since all traffic sources convert similarly, optimize based on acquisition costs
- Redistribute marketing budget toward most cost-effective channels



# Agenda

- Block 1 - Build & Manage a Data Infrastructure – Kayak Project
- Block 2 – Exploratory Data Analysis – Steam Project
- Block 3 – Supervised Machine Learning – Conversion Project
- Block 3 – Unsupervised Machine Learning – Uber Pickups Project
- Block 4 – Deep Learning – AT&T Spam Detection Project
- Block 5 – Deployment – Getaround Project
- Block 6 – Lead a Data Project – Offensive Speech Recognition



# Project Reminder



## Project

- One of the main pain point that Uber's team found is that sometimes drivers are not around when users need them.
- Therefore, Uber's data team would like to work on a project where their app would recommend hot-zones in major cities to be in at any given time of day.

## Goal

The target of the project is to

- Develop an algorithm to identify "hot zones" where drivers should position themselves
- Create time-based recommendations that adapt to changing demand patterns
- Visualize results for easy implementation by drivers



# The dataset represents latitude and longitude of 564k pickups in April 2014

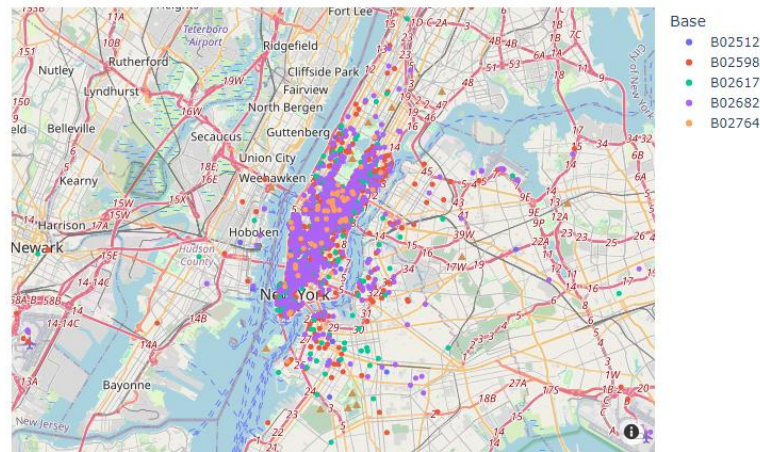
Uber

## Dataset description

- Dataset of April 2014 used with 564k lines
- 4 columns:
  - Date
  - Latitude
  - Longitude
  - Base : Internal code, not used in the analysis
- Preprocessing limited to converting the date column into several sub-columns
- Focus on New York City inside this latitude and longitude line :
  - Latitude minimum = 40.4774
  - Latitude maximum = 40.9176
  - Longitude minimum = -74.2591
  - Longitude maximum = -73.7004
- Analysis done on the 30<sup>th</sup> of April 2014 at 5pm to limit the number of lines

## Overview of the pickup location for a given hour of a given day

Pick up of the 30th at 5pm

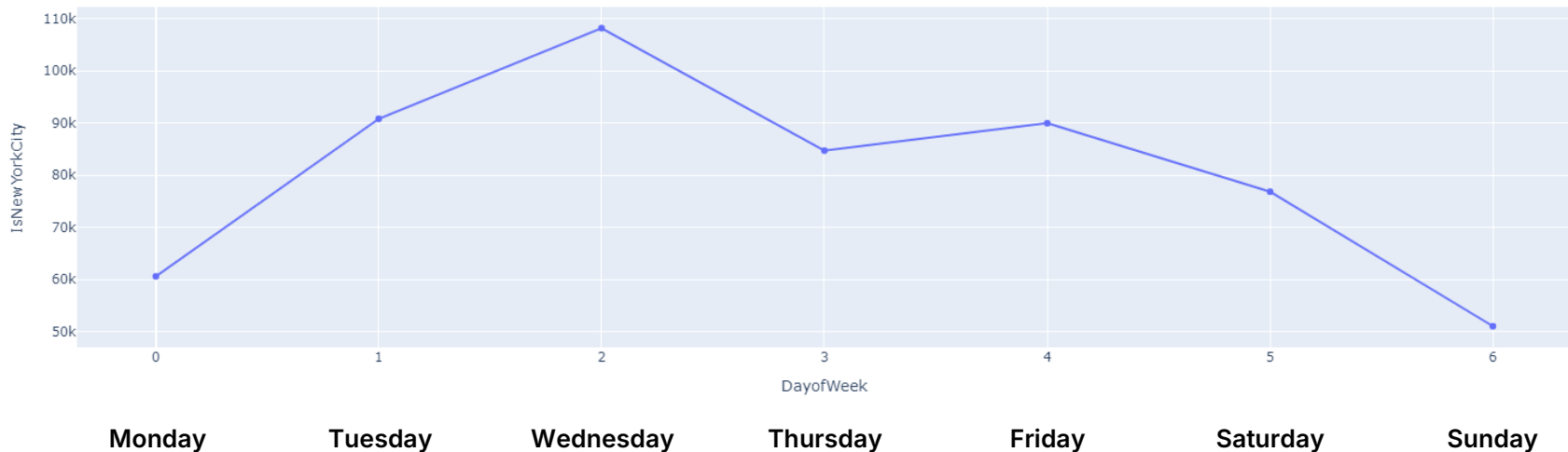




# Monday and Sunday are the lowest day while the peak in on Wednesday

Uber

Number of pickups per Day | April 2014, NYC only, 0 = Monday

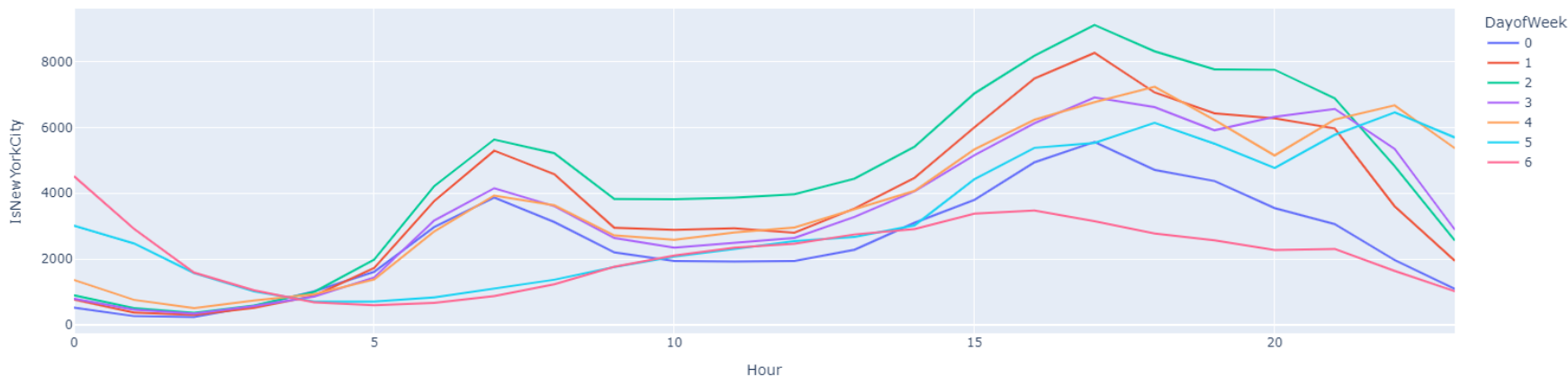




During the week, there is a peak at 7am then between 5pm and 8pm while the night is busy during the weekend



Number of pickups per Hour depending on the day | April 2014, NYC only, 0 = Monday







# DBScan is used to calculate coordinates of hot zones at any given time

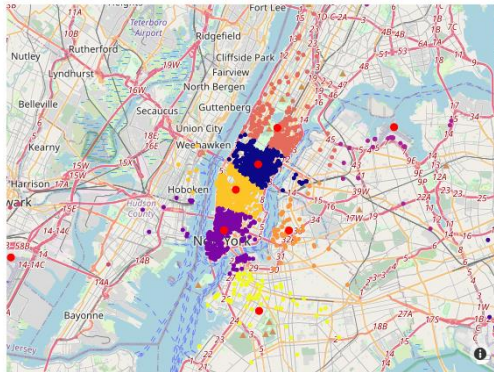
Uber

## Kmeans Clustering

### Description

- Elbow methodology and inertia calculation to get the optimal number of clusters
- 9 clusters seems to be the best for April 30<sup>th</sup>

### Cluster Overview

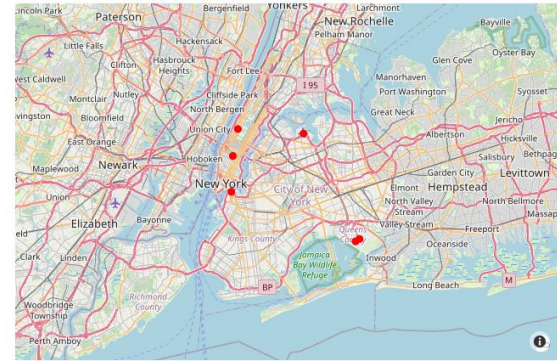


## DBScan

### Description

- DBScan used to handle different numbers of cluster depending on the time
- Parameters : Epsilon = 0.1 / Min Sample = 10
- 6 clusters + outliers

### Cluster Center Overview



DBScan chosen for algorithm as the number of cluster adapts to the dataset



# Hot zone are calculated and plotted for any given time with a DBScan clustering

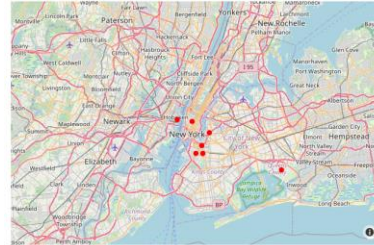
Uber

## Plot\_hot\_zone function

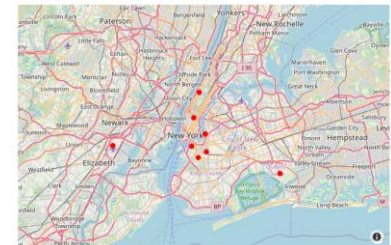
- Input : dataset, day of the week, hour, dbscan parameters
- For any given hours, would calculate the clusters center and plot them
- The output would be a map with the hot zones of this hour
- For a given day, the evolution of hot spots can be shown by looping over different hours

## Evolution of hot zones for a Saturday at 12am, 6am, 12pm, 6pm

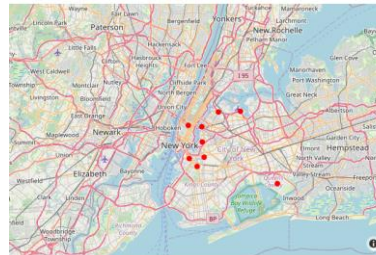
12am



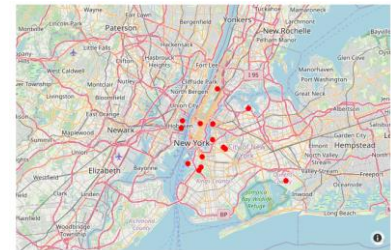
6am



12pm



6pm





# Agenda

- **Block 1 - Build & Manage a Data Infrastructure – Kayak Project**
- **Block 2 – Exploratory Data Analysis – Steam Project**
- **Block 3 – Supervised Machine Learning – Conversion Project**
- **Block 3 – Unsupervised Machine Learning – Uber Pickups Project**
- **Block 4 – Deep Learning – AT&T Spam Detection Project**
- **Block 5 – Deployment – Getaround Project**
- **Block 6 – Lead a Data Project – Offensive Speech Recognition**



# Project Reminder



## Project

- One of the main pain point that AT&T users are facing is constant exposure to SPAM messages.
- AT&T has been able to manually flag spam messages for a time, but they are looking for an automated way of detecting spams to protect their users.

## Goal

Your goal is to build a spam detector, that can automatically flag spams as they come based solely on the sms' content.



# Transforming sentence to encoded list to feed the model



## Dataset description

- Data structure
  - 1 target column : spam or ham (not spam)
  - 3 column with truncated messages
  - All 3 columns merges to get the total sentence
- 5,572 lines
- 13.41% of spam message

## Data Preprocessing

### Data cleaning

- Removing non-alphanumeric characters
- Converting to lower case

### Spacy Cleaning

- Lemmatization of each sentence
- Stop Words removal

### Tokenization and padding

- Tokenization : Conversion of each sentence to a list of number
- Padding : Scaling of each sentence to a 100-element list

|   | target | complete_text                                     | complete_text_clean                               | text_encoded                                      | len_review |
|---|--------|---|---|---|------------|
| 0 | 0      | Go until jurong point, crazy.. Available only ... | jurong point crazy available bugis n great wor... | [3621, 233, 447, 462, 944, 35, 52, 205, 945, 7... | 14         |
| 1 | 0      | Ok lar... Joking wif u oni...                     | ok lar joke wif u oni                             | [10, 194, 463, 289, 1, 1459]                      | 6          |
| 2 | 1      | Free entry in 2 a wkly comp to win FA Cup fina... | free entry 2 wkly comp win fa cup final tkts 2... | [12, 298, 3, 534, 666, 33, 1460, 857, 423, 146... | 22         |
| 3 | 0      | U dun say so early hor... U c already then say... | u dun early hor u c                               | [1, 125, 150, 2362, 1, 85]                        | 6          |
| 4 | 0      | Nah I don't think he goes to usf, he lives aro... | nah think usf live                                | [711, 22, 667, 129]                               | 4          |



# A neural network with GRU layers were chosen for deployment



|                 | Description   | Results  |
|-----------------|---|--|
| Embedding Model | <ul style="list-style-type: none"><li>• 1 Embedding layer</li><li>• Dropout / Dense final layers</li></ul>  | <ul style="list-style-type: none"><li>• F1 Score : 0.94</li><li>• Recall : 0.89</li><li>• Accuracy : 0.98</li></ul>                            |
| GRU             | <ul style="list-style-type: none"><li>• Gated recurrent units neural networks</li><li>• 1 Embedding layer</li><li>• 1 GRU layers with 32 units</li><li>• Max Pooling / Dropout / Dense final layers</li></ul> | <ul style="list-style-type: none"><li>• F1 Score : 0.94</li><li>• Recall : 0.93</li><li>• Accuracy : 0.98</li></ul>                            |
| LSTM            | <ul style="list-style-type: none"><li>• Long short-term memory neural network</li><li>• 1 Embedding layer</li><li>• 1 LSTM layer with 16 units</li><li>• Max Pooling / Dropout / Dense final layers</li></ul> | <ul style="list-style-type: none"><li>• F1 Score : 0.94</li><li>• Recall : 0.93</li><li>• Accuracy : 0.98</li></ul>                            |
| Glove + LSTM    | <ul style="list-style-type: none"><li>• Use of pre-trained embedding matrix</li><li>• LSTM layers with 128 units</li><li>• Max Pooling / Dropout / Dense final layers</li></ul>                               | <ul style="list-style-type: none"><li>• F1 Score : 0.85</li><li>• Recall : 0.74</li><li>• Accuracy : 0.96</li></ul>                            |
| BERT            | <ul style="list-style-type: none"><li>• Use of BERT pre-trained model from 2018</li><li>• Use of preprocess and encoder BERT layers</li><li>• Dropout and Dense final layers</li></ul>                        | <ul style="list-style-type: none"><li>• F1 Score : 0.93</li><li>• Recall : 0.89</li><li>• Accuracy : 0.98</li><li>• 3h training time</li></ul> |

Best  
Performance



# Agenda

- Block 1 - Build & Manage a Data Infrastructure – Kayak Project
- Block 2 – Exploratory Data Analysis – Steam Project
- Block 3 – Supervised Machine Learning – Conversion Project
- Block 3 – Unsupervised Machine Learning – Uber Pickups Project
- Block 4 – Deep Learning – AT&T Spam Detection Project
- Block 5 – Deployment – Getaround Project
- Block 6 – Lead a Data Project – Offensive Speech Recognition



# Project Reminder

getaround

## Project

- A key operational challenge identified by Getaround is the issue of late returns at checkout, which can cause significant problems:
- To address these issues, Getaround is implementing a minimum delay between consecutive rentals, preventing cars from appearing in search results if requested checkin/checkout times are too close to existing bookings.

## Goal

This project has several objectives:

- **Optimal Minimum Delay Threshold:** Identify the ideal time buffer between consecutive rentals that balances customer satisfaction with revenue impact.
- **Pricing Optimization:** Build a machine learning model to predict the optimal rental price based on car features and rental patterns.
- **Data Analysis App:** Create an interactive Streamlit dashboard to visualize the main analysis and simulate the impact of different threshold settings on rental availability and revenue.
- **API:** Create a RESTful API to make the pricing model accessible for production use, allowing for seamless integration with other systems.





# Key Findings



**Short time gaps between reservations represent a minor portion of business operations**

- Out of 21k rentals, only 8% have a time gap below 12 hours between consecutive rentals
- On average, each car is rented fewer than 3 times, indicating moderate utilization
- Less than 400 rentals (approximately 2%) have a time gap below 1 hour from the previous rental

**Late checkouts have limited impact on overall business operations:**

- Only 218 rentals were affected by late checkouts, where the car was not available at the scheduled time
- The cancellation rate for affected rentals is around 17%, which is comparable to the average cancellation rate of 15%
- Most delays were under 30 minutes, likely due to minor traffic issues, which wouldn't typically justify a cancellation

**A buffer of 30-60 minutes between rentals appears sufficient to minimize scheduling conflicts**

- Given the current rental frequency, aggressive time optimization does not appear necessary
- Most delays are less than 1 hour, and this buffer would prevent most potential issues
- Approximately 2% of reservations would be affected by implementing this threshold



# 21k rentals, 80% on mobile and 15% cancellation rate

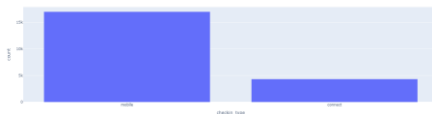
getaround

## Dataset description

- There are 7 columns with 21,310 lines :
  - Rental ID
  - Car ID
  - checkin type : mobile or connect
  - state : canceled or ended
  - delay\_at\_checkout\_in\_minutes
  - previous\_ended\_rental\_id : Only 9% data available on previous rental
  - time\_delta\_with\_previous\_rental\_in\_minutes
- Hypothesis : Let's assume that NaN is meaning that the previous rental was more than 12 hours before or that there were no previous rental

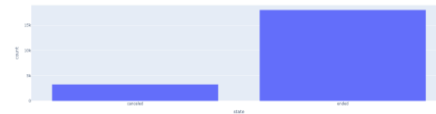
## Distribution of main columns of the dataset

Checkin type



**80% mobile rental**

State



**15% cancellation**



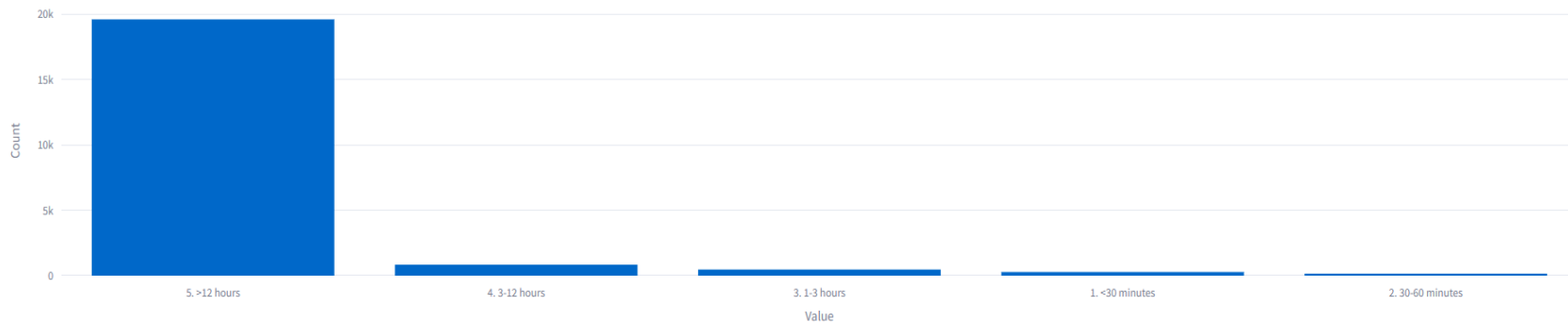
## Short time gaps between reservations represent a minor portion of business operations

getaround

### Distribution of rental by time vs. previous rental |

When not data, it is assumed that delay is higher than 12 hours

Distribution of time\_vs\_previous\_rental\_category



|      |        |     |     |     |     |
|------|--------|-----|-----|-----|-----|
| In # | 19,599 | 841 | 469 | 279 | 122 |
| In % | 92%    | 4%  | 2%  | 1%  | 1%  |



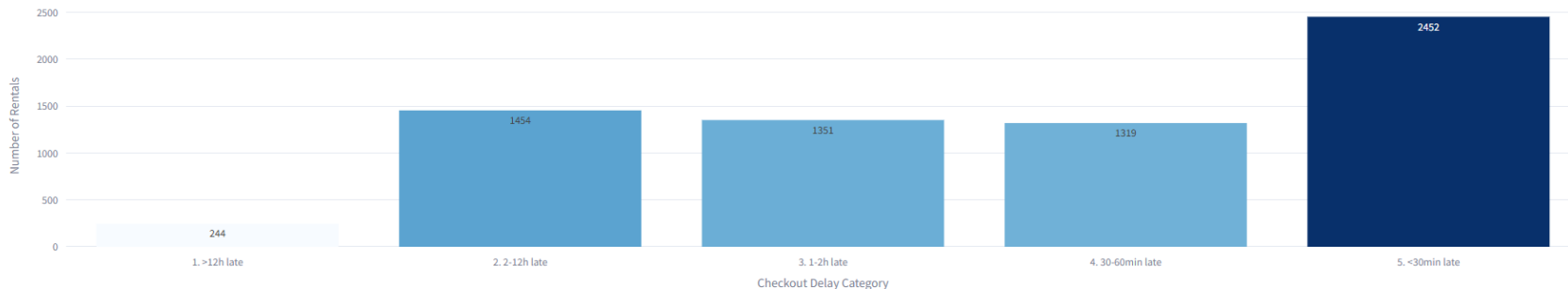
# Main delay is below 30 minutes and majority is below 60 minutes

getaround

## Distribution of late checkout by time of delay|

---

Number of Late Rentals by Checkout Delay Category





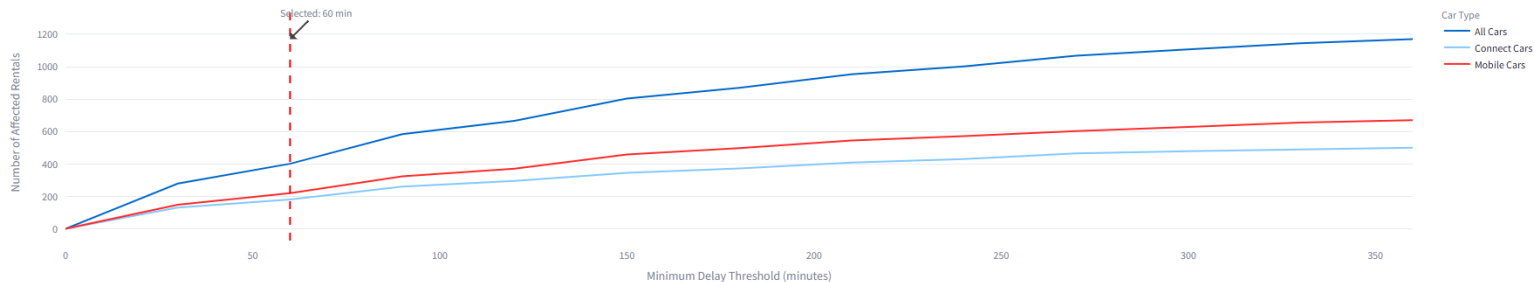
60 minutes seems a good threshold given the low number of rentals impacted

getaround

## Distribution of late checkout by time of delay

### Impact of 60-minute Minimum Delay

Number of Affected Rentals by Threshold



**181 rentals would be impacted with a threshold of 60 minutes**



# Streamlit App : 4 tabs for detailed analysis



## Streamlit app overview



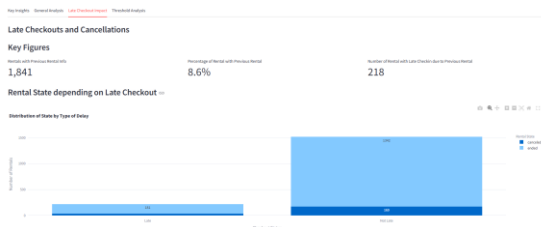
### Tab 1 : Key Insights



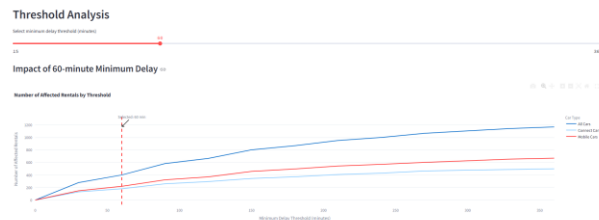
### Tab 2 : General Analysis



### Tab 3 : Late Checkout Impact



### Tab 4 : Threshold Analysis





## 4.8k lines of rental data to get a pricing prediction

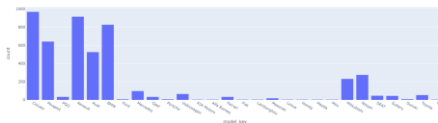
getaround

### Dataset description

- 4,843 lines in the dataset and no missing data
- 15 columns
  - Car categorical information : model, fuel, color, type,
  - Car numerical information : mileage, engine power
  - Boolean information : GPS, air conditioning, automatic, connect, speed regulator, winter tires
  - target : rental price
- Limited data cleaning
  - Removal of unused column
  - Flag as 'Others' the model with less than 5 occurrence in the dataset

### Distribution of some columns of the dataset

Model



Mileage



Fuel



Price





# Different Catboost model tested on MLFlow for a final R2 test of 0.78

getaround

mlflow

## Model used

### Preprocessing

- Numerical data : Standard Scaler
- Categorical : One Hot Encoder

### Model

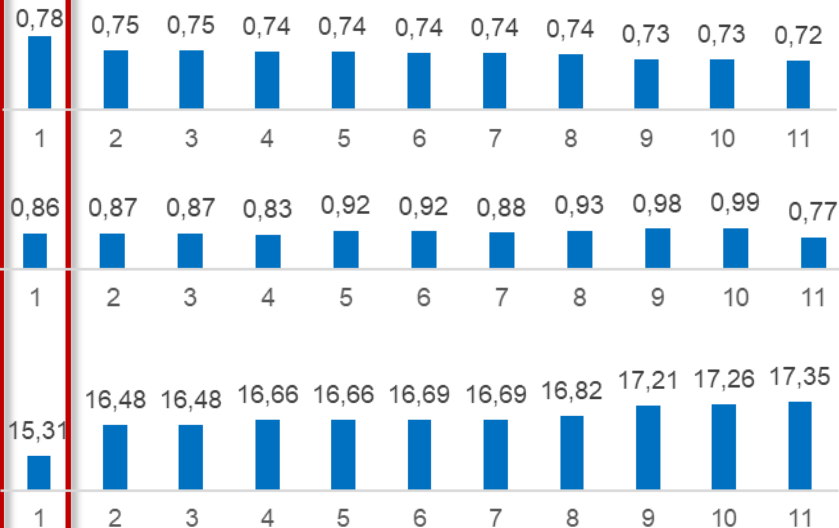
- Model used : Catboost Regressor
- Different experiments done on MLFlow
- Best model used for the API
- Parameters :
  - Depth
  - Iterations
  - Learning Rate

## Experiments overview

R2 Test

R2 Train

F1 Score



Best model registered in MLFlow



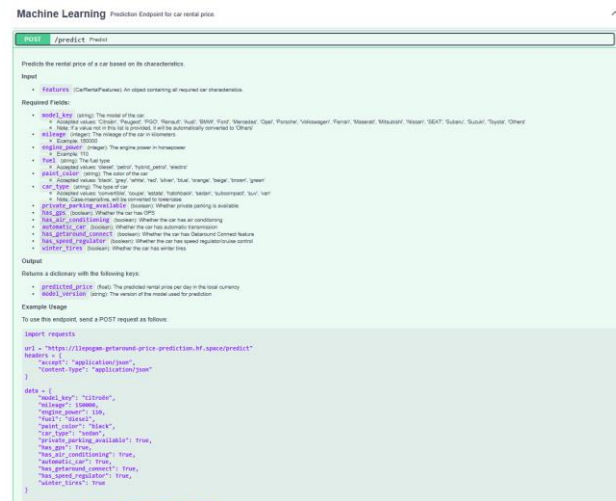


## Streamlit app overview

### Description

- Text input from the user
  - 13 value to enter
  - validation of data format
  - if model unknown, replace by Others
- Use of registered model stored in MLFlow
- API stored in Huggingface

## Predict Endpoint





# Agenda

- Block 1 - Build & Manage a Data Infrastructure – Kayak Project
- Block 2 – Exploratory Data Analysis – Steam Project
- Block 3 – Supervised Machine Learning – Conversion Project
- Block 3 – Unsupervised Machine Learning – Uber Pickups Project
- Block 4 – Deep Learning – AT&T Spam Detection Project
- Block 5 – Deployment – Getaround Project
- Block 6 – Lead a Data Project – Offensive Speech Recognition



“Moderation is a propaganda  
word for censorship”

Elon Musk, interview with Don Lemon, March 2024



# Big tech companies are reducing their focus on moderation leading to increase of hate speech

TECHNOLOGY AND THE INTERNET

One billionaire owner, twice the hate: Twitter hate speech surged with Musk, study says



Twitter Chief Executive Elon Musk speaks at a marketing conference in Miami Beach, Fla., on April 18, 2023. (Chamber Brown / AFP/Getty Images)

By Christian Martinez

#### Subscribers are Reading >

Los Angeles has never seen this level of destruction: 'Everything is burned down' >

Among tens of thousands of displaced Angelenos, celebrities face the same devastating losses

Why hydrants ran dry as firefighters battled California's deadly fires

These are the Malibu and Altadena restaurants damaged or destroyed by ongoing L.A. fires

Before-and after satellite images show destruction in Malibu and Altadena

#### Latest Technology and the Internet

Web 3.0, an old, wildfire app Watch Duty adds 600,000 users overnight

Jan. 8, 2023



Meta's new hate speech guidelines permit users to say LGBTQ people are mentally ill

Changes to its hate speech guidelines were among broader policy shifts Meta made to its moderation practices.





# Objectives : build a fully deployed offensive speech detection model

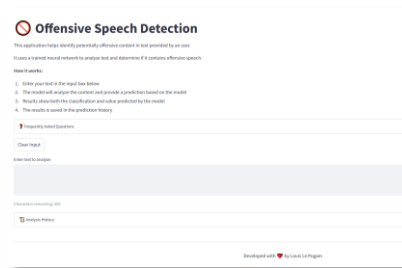
## Target

Build a model to  
detect offensive  
speech








## Results

Deployed model with  
73% accuracy and  
68% recall





# 4 steps to get a deployed model

|                  | Description  | Technology  |
|------------------|--|---|
| 1<br>Dataset     | <ul style="list-style-type: none"><li>Choose dataset to use in the training</li></ul>  |  <b>Hugging Face</b>   |
| 2<br>Model       | <ul style="list-style-type: none"><li>Build preliminary models to define the best design</li></ul>   |    |
| 3<br>Fine-Tuning | <ul style="list-style-type: none"><li>Fine-tune the model to find best parameters</li><li>Register prediction and preprocessing models</li></ul> |    |
| 4<br>Deployment  | <ul style="list-style-type: none"><li>Make the model available through an API</li><li>Create an app with Streamlit to use it</li></ul>           |  <b>FastAPI</b>  <b>Streamlit</b> |



# OLID dataset used for training, with 13.2k tweets, of which 33.2% are offensive

## Source

- Catalog of abusive language data (PLOS 2020)
- Paper : Vidgen B, Derczynski L (2020) *Directions in abusive language training data, a systematic review: Garbage in, garbage out*
- Catalogue of datasets in different languages
- 25 languages available
- 59 datasets in English

## Dataset used

- Name : The Offensive Language Identification Dataset (OLID)
- Paper : *Predicting the Type and Target of Offensive Posts in Social Media (2019)*
- Available data : Annotated tweets flagged offensive or not offensive
- Size : 13,240 tweets, of which 4,400 flagged as offensive (33.2% of total)
- Available on Hugging Face

| tweet   | subtask_a |
|---|-----------|
| @USER She should ask a few native Americans wh... | OFF       |
| @USER @USER Go home you're drunk!!! @USER #MAG... | OFF       |
| Amazon is investigating Chinese employees who ... | NOT       |
| @USER Someone should'veTaken" this piece of sh... | OFF       |
| @USER @USER Obama wanted liberals & amp; illeg... | NOT       |



# A neural network with GRU layers were chosen for deployment

## Description

## Preprocessing

## Results

### GRU

- Gated recurrent units neural networks
- 1 Embedding layer
- 1 GRU layers with 64 units
- Max Pooling / Dropout / Dense final layers

- Text cleaning : Punctuation and symbols removal, conversion to lowercase
- Lemmatization and Stop Words removal
- Encoding and Padding

- F1 Score : 0.63
- Recall : 0.66
- Accuracy : 0.74

### Chosen model

### LSTM

- Long short-term memory neural network
- 1 Embedding layer
- 2 LSTM layers with 64 units
- Max Pooling / Dropout / Dense final layers

- Text cleaning : Punctuation and symbols removal, conversion to lowercase
- Lemmatization and Stop Words removal
- Encoding and Padding

- F1 Score : 0.61
- Recall : 0.58
- Accuracy : 0.75

### BERT

- Use of BERT pre-trained model from 2018
- Use of preprocess and encoder BERT layers
- Dropout and Dense final layers

- Text cleaning : Punctuation and symbols removal, conversion to lowercase
- Lemmatization and Stop Words removal
- BERT pre-trained preprocessed layer using directly text

- F1 Score : 0.59
- Recall : 0.57
- Accuracy : 0.73
- 3h training time





# The model registered in MLFlow reached 68% recall and 73% accuracy

## MLFlow Experiments



## Results

### Overview

- Change the number of units of each layers
- Add intermediate layers
- Increase or decrease the number or epochs for training

### Experiments

hate\_speech\_detection [Provide Feedback](#) [Add Description](#)

Run Evaluation [Experiment](#) [Trains](#) [Experiment](#)

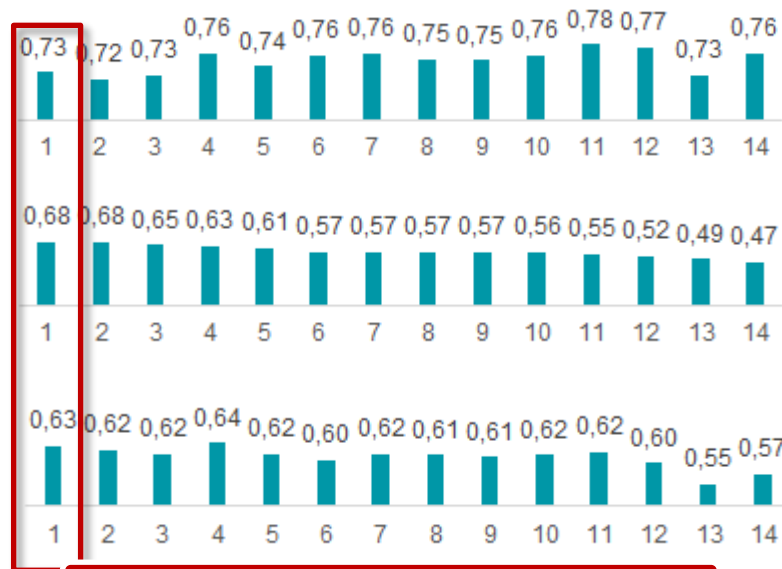
metrics:acc = 1 and param:epochs = 100

| Run Name           | Created    | Dataset | Duration |
|--------------------|------------|---------|----------|
| powerful-shark-573 | 3 days ago | -       | 7.5sec   |
| powerful-shark-600 | 3 days ago | -       | 6.8sec   |
| powerful-shark-514 | 3 days ago | -       | 4.7sec   |
| powerful-shark-600 | 3 days ago | -       | 9.5sec   |
| bold-hug-410       | 1 day ago  | -       | 7.2sec   |
| spring-rabbit-239  | 3 days ago | -       | 6.8sec   |
| musical-bear-100   | 3 days ago | -       | 4.6sec   |
| musical-bear-36    | 3 days ago | -       | 5.5sec   |
| musical-bear-700   | 3 days ago | -       | 7.5sec   |
| musical-shrimp-95  | 1 day ago  | -       | 7.5sec   |
| musical-bear-398   | 3 days ago | -       | 4.7sec   |
| musical-bear-684   | 3 days ago | -       | 5.5sec   |
| musical-bear-641   | 3 days ago | -       | 6.7sec   |
| musical-bear-185   | 3 days ago | -       | 9.5sec   |
| musical-bear-192   | 4 days ago | -       | 5.5sec   |
| musical-bear-221   | 9 days ago | -       | 4.7sec   |

### Accuracy

### Recall

### F1 Score

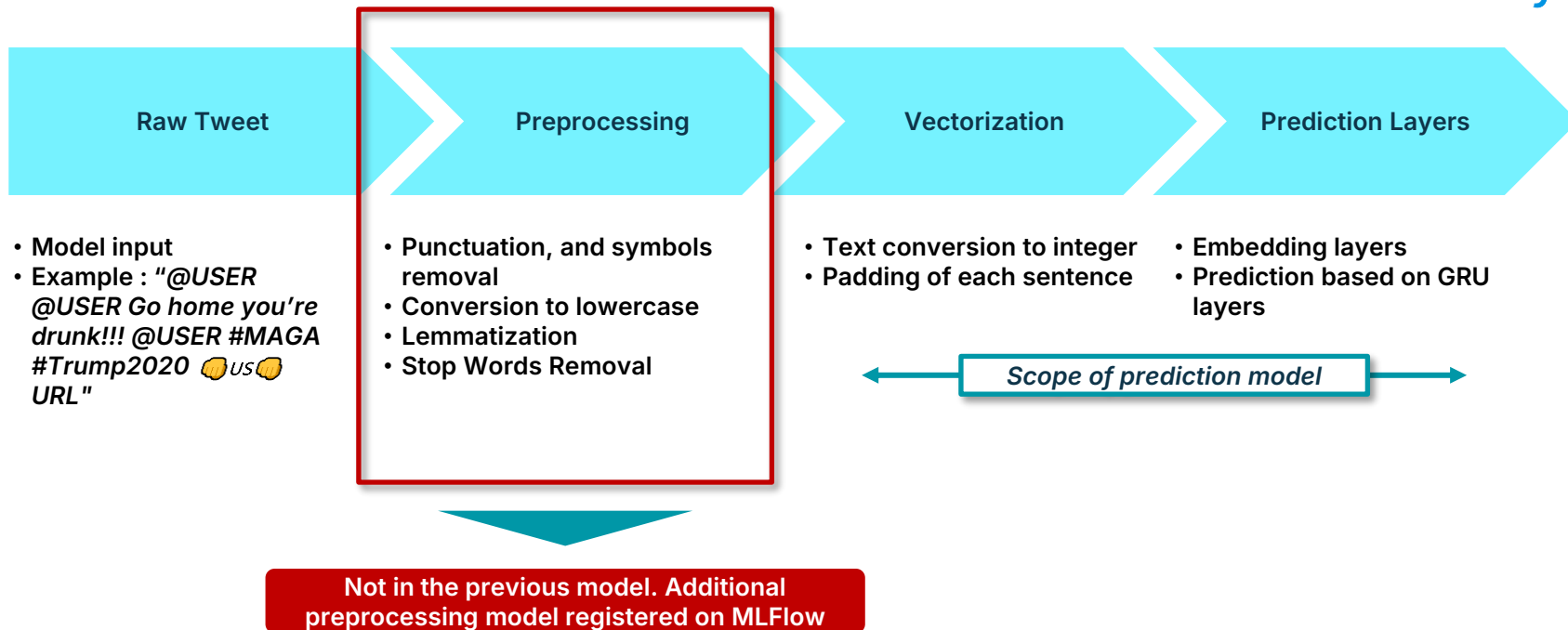


Model with highest recall registered in MLFlow



# Additional preprocessing model was registered on MLFlow

mlflow





# The model were deployed through a Streamlit app use an API build with Fast API

## API



### Description

- FastAPI API deployed on Hugging Face
- 2 endpoints using MLFlow models :
  - /preprocessing
  - /predict using the preprocessing model
- Prediction output
  - Prediction : "Offensive" or "Not Offensive"
  - Probability : Number between 0 and 1, the higher, the more offensive

### Overview



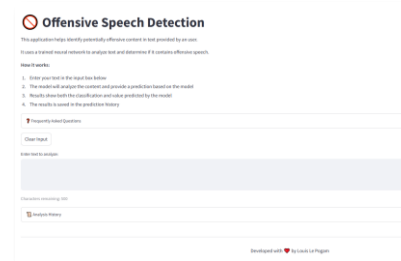
## Detection App



### Description

- Streamlit app deployed on Hugging Face
- Text entered as an input from the user
- Output from the model shown after validation
- Various explanation and FAQ
- History of research saved on a S3 bucket

### Overview





# Demonstration

Let's open the app and see how it works!



# Next Steps : Further improve the model and the app

## Improvements

---

### Model

- Train the model on bigger dataset to improve the performance
- Use remote training to use more ambitious models
- Leverage on improvements in text detection and pre-trained model

### App Performance

- Improve app performance to insure quicker prediction
- Additional costs required to have access to better server

### App UI

- Handle better exception and potential errors
- Add additional endpoint and features (e.g. batch predict from a csv file)



# Thanks!

