Laboratorio di web scraping
AA. 2024-2025

docente: Laura Ricci

laura.ricci@unipi.it

Lezione 1
introduzione al corso
29/01/2025

INFORMAZIONI GENERALI

- Docente: Laura Ricci, laura.ricci@unipi.it
- Crediti: 6
- Link alla pagina Moodle del corso: https://elearning.di.unipi.it/course/view.php?id=1030
- Quando e dove:

  14:00 -16:00 Mercoledì, AULA C

  11:00 -13:00 Venerdì, AULA C

- Disponibile per:
  - laurea triennale in informatica – nuovo ordinamento
  - laurea triennale in informatica – vecchio ordinamento (crediti liberi)
- Prerequisiti:
  - Programmazione
  - Algoritmica
  - Basi di dati
  - Reti

MODALITA' DI ESAME

- modalità di esame: progetto + orale
- progetto di fine corso
  - consegnato all'inizio di `Aprile 2025`
  - valido fino alla fine di `Aprile 2026`
  - possibilità di proporre argomenti di progetto, che comunque dovranno essere approvati
- prova orale include
  - discussione del progetto
  - domande su argomenti del corso non coperti dal progetto
- esame, su appuntamento, circa una settimana dopo la consegna del progetto.
  - ma occorre registrarsi e compilare il questionario entro le date che saranno pubblicate su Alice!

PROGRAMMA PRELIMINARE: STRUMENTI DI BASE

- *Python*
  - introduzione mediante esempi
- *reperire i dati dal web:*
  - *Web Scraping*
    - analisi di pagine statiche : `BeautifulSoup`
    - analisi di pagine dinamiche : `Selenium`
  - *API*: user authentication, costruzione di queries, esempi: `Google Big Queries`
- la libreria `Pandas`:
  - `DataFrame` e `Series`: selezione di elementi per indice e per condizioni
  - daggruppare elementi: `Group_by`
  - merging, concatenating, e pivoting
  - visualizzazione: `Matplotlib, seaborn`

PROGRAMMA PRELIMINARE: ANALISI DEI DATI

- *Statistica descrittiva*
  - descrizione delle caratteristiche di una distribuzione dei dati
  - metriche classiche, analisi temporali
  - valutazione di diverse modalità di visualizzazione
- *Passi e strumenti necessari*
  - reperire dati: scraping, `API`
  - eventuale parsing dei dati
  - data cleaning (dati incompleti, errori, …)
  - scelta della struttura dati per la memorizzazione
    - `Pandas`
    - `Database`
  - analisi dei dati

PROGRAMMA PRELIMINARE: TECNICHE AVANZATE

- *Statistica inferenziale*
  - utilizzare un campione dei dati per stimare caratteristiche della popolazione, o per testare una ipotesi sull'intera popolazione
- *Supervised Learning*
- *Data Clustering*
  - individuare la struttura intrinseca dei dati, senza una fase apprendimento
  - `K-means, Principal Component Analysis`
- *Network Analysis*
  - proprietà caratteristiche: diametro, centrality, clustering coefficient, communities
  - graph models: random graphs, scale free networks, small worlds.
- obiettivo: illustrare l'applicazione di almeno una delle tecniche precedenti
  - network analysis
  - data clustering?

DISCLAIMER

- il titolo del corso può essere fuorviante: il corso non tratta solamente di web scraping!
- `Web Scraping` è una parte del corso, ma il corso, come risulta chiaro dalle slide precedenti si occupa di
  - reperimento di dati, con tecniche scraping, ma
  - analisi dei dati reperiti
- complesso dal punto di vista burocratico cambiare il titolo
- per capire se il corso è di interesse considerare il programma presentato nelle slide precedenti

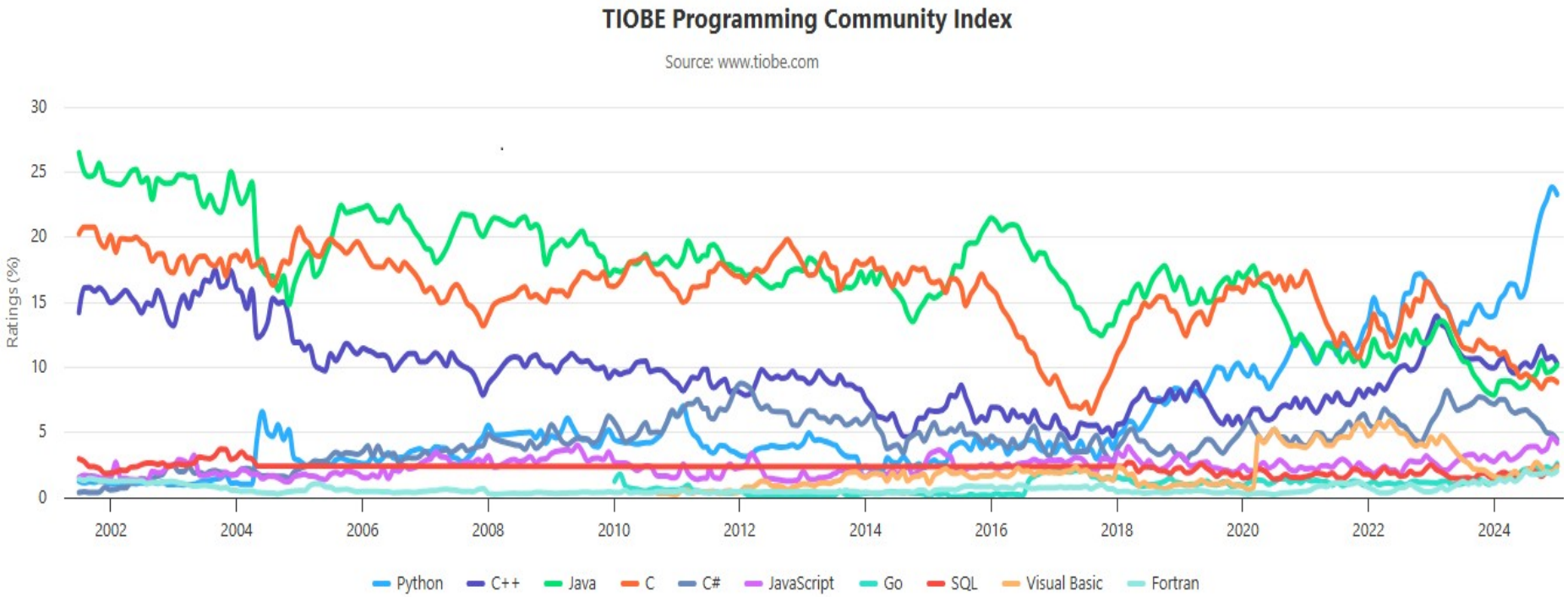
CASI D'USO

- *e-commerce:* lancio di nuovi prodotti
  - analisi dei prezzi sui siti di e-commerce
  - reviews e rating effettuati dagli utenti per prodotti simili
- *viaggi*
  - confronto di dati provenienti da siti diversi
  - frequente fluttuazione di prezzo, dinamicità
  - API di siti coma `Trivago`, interessanti per studio di tecniche antiscraping
- *blockchain e cryptocurrencies*
  - transazioni pubbliche, nessun problema di privacy.
  - struttura delle transazioni (`Bitcoin`, `Ethereum`)
  - grafi di transazioni
  - scraping dal siti web dell'ecosistema delle blockchain: blockchain exploreres, `NFT` Markets, exchangers
  - Merge dei dati provenienti da transazione + scraping

CASI D'USO

- *contatti commerciali:*
  - ricerca in siti web di potenziali contatti per il target market
  - recruiting industry
    - scraping di candidati per un potenziale lavoro
    - `Linkedin` mette in atto molte politiche anti scarping, soluzioni alternative
- *social media*
  - metriche come like, shares, comment, possono essere molto importanti per identificare trend e possibilità di investimenti
  - problema: forti politiche antiscraping
    - `Twitter API` completamente aperta fino a `Novembre 2022`
    - `Linkedin`, `Facebook`, attualmente molto "chiusi"
    - alternativa: distributed social networks: `Mastodon`, `Bluesky`

I LINGUAGGI PIU' POPOLARI (GENNAIO 2025)



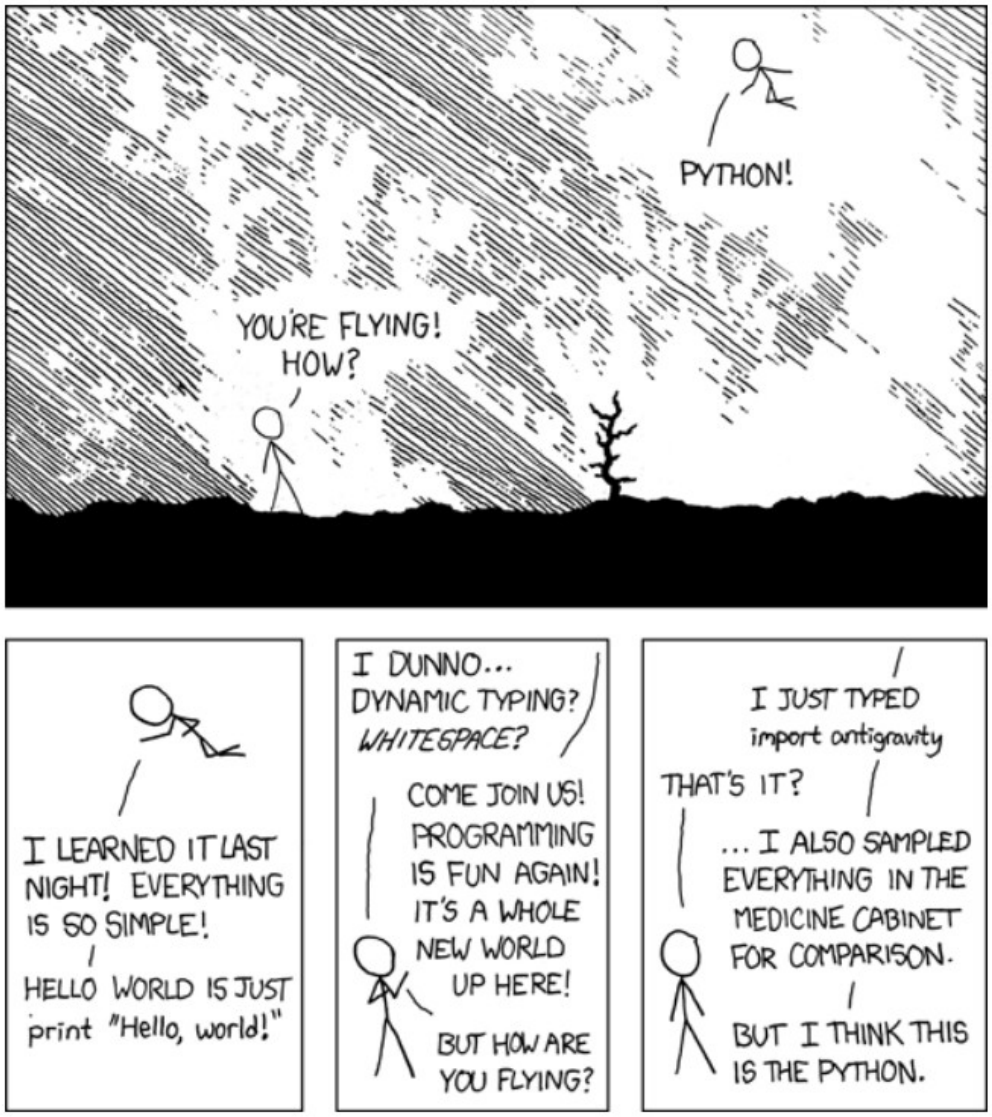
- Python è attualmente il linguaggio più utilizzato
- permanentemente tra i primi 10 dal 2002
- netto sorpasso su tutti gli altri linguaggi nel 2023

PYTHON: BREVE STORIA

- pensato come successore del linguaggio ABC
- prime linee di codice nel 1989 da `Guido van Rossum` (`Univ. of Amsterdam`)

  "*Python is an interpreted, interactive, object-oriented programming language. Python combines remarkable power with very clear syntax. It has modules, classes, exceptions, very high level dynamic data types, and dynamic typing*"
- il nome è ispirato da "`Monty Python's Flying Circus`"...
- `Python 2.0: Ottobre 2000`
- `Python 3.0: Dicembre 2008`

PERCHE' PYTHON?

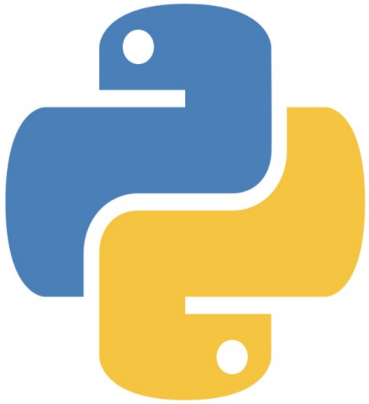


TIM PETERS: THE ZEN OF PYTHON

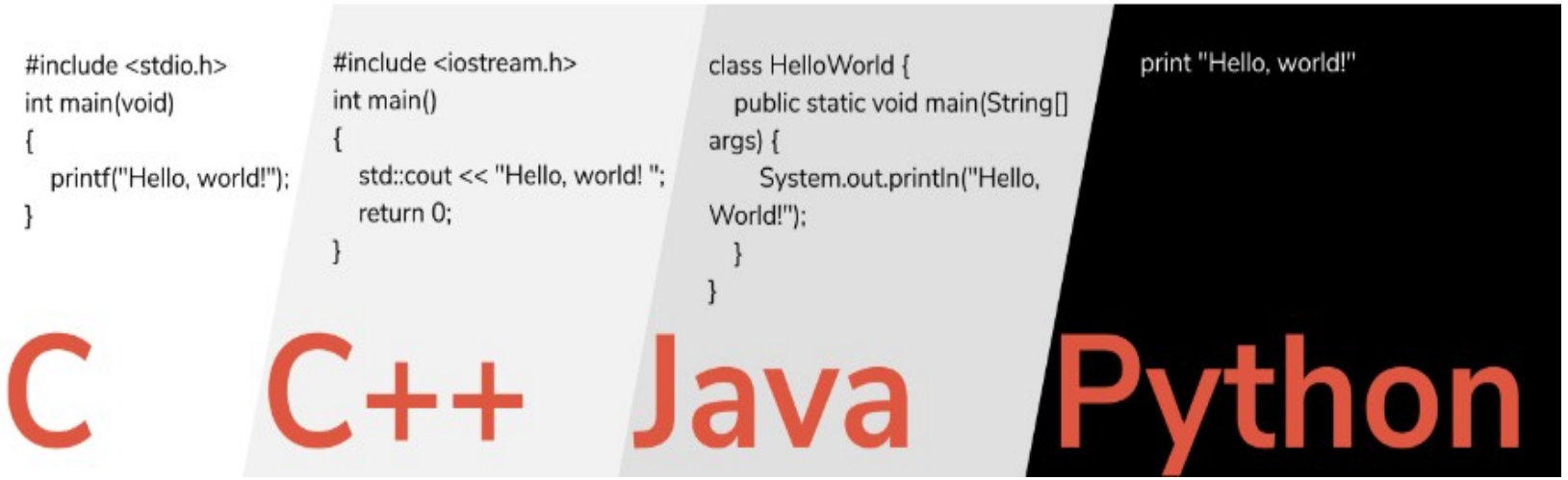
```
Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
```

PERCHE' PYTHON?

- general purpose, interpretato
- indipendente dalla piattaforma: il byte code è eseguito dall'interprete `Python`
- estremamente versatile
  - web programming, data analysis, numerical analysis,...
- sintassi semplice, espressiva (quasi pseudo-codice)
- dynamic and strong typing
- ricchezza di librerie
  - *"do not reinvent the wheel!"*
- una ricca community on-line
- svantaggi
  - lentezza
  - ...ma facilmente "extensible e embeddable": molte parti scritte in `C/C++`

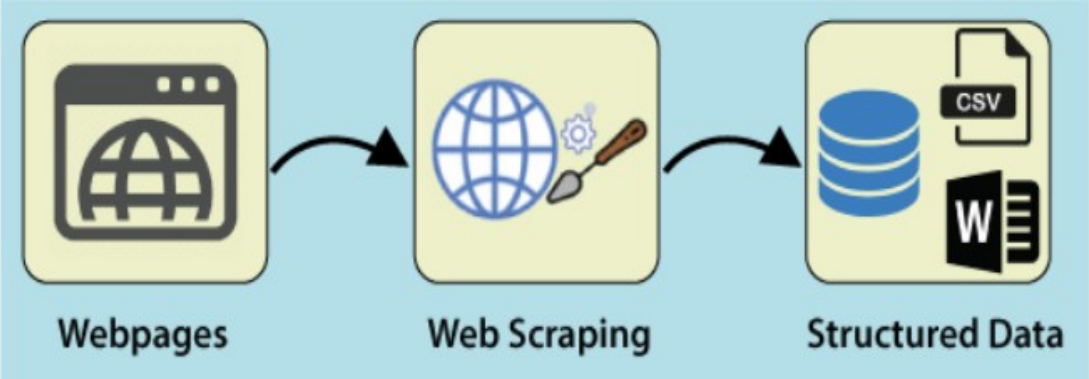


PYTHON: "HELLO WORLD"



- adatto per la data science e, più in generale, per lo scienziato che deve analizzare i dati di un esperimento, perchè fornisce funzionalità per:
  - reperire i dati (da web, da simulazioni, dati sperimentali,...)
  - trasformare e elaborare i dati
  - visualizzare i risultati in formati il più possibile intuitivi e di alta qualità

REPERIMENTO DEI DATI: WEB SCRAPING



- *"web scraping (also referred as web harvesting) is defined as "the practice of gathering data through any means other then a program interacting with an API (or, obviously, through a human using a web browser)"*
- *"usually done by writing a software program that automatically extracts data from one or more websites, first it queries a web server, requests data (fetching an HTML file) and then parses that data to extract the desired information.*
- sinonimi: web scraper, crawler, spider o semplicemente bot
- `Python` fornisce lo stesso ambiente sia per lo scraping che per l'analisi dei dati strutturati reperiti

WEB SCRAPING: MA E' LEGALE?

- la situazione in Italia: molta attenzione, soprattutto di recente, da parte del Garante della Privacy, per quanto riguarda la raccolta di dati personali per il training degli algoritmi di intelligenza artificiale
- attenzione sulla normativa sulla data protection (GDPR)
- caso Clearview AI, sanzionata dal Garante della Privacy
  - sviluppato un motore di ricerca per il riconoscimento facciale
  - raccolti, mediante scraping, miliardi di immagini di volti di persone estratti da siti web in cui vi sono foto pubblicamente accessibili
  - elaborazione delle immagini con tecniche biometriche, trasformazione ed indicizzazione
  - motore di ricerca per immagini
- scraping non legale nel caso di dati personali: violazione di un articlo del GDPR
  - grossa sanzione alla società americana

WEB SCRAPING: MA E' LEGALE?

- riconosciuta la legalità in U.S.A. (2022)
- huQ Labs raccoglieva informazioni dai profili Linkedin
  - riteneva scraping non illegale se i dati sono pubblicamente accessibili
  - i profili Linkedin venivano utilizzati per costruire predizioni sulla probabilità che un dipendente lasci una azienda
  - diffida di Linkedin a huQ Labs
    - Linkedin riteneva tale attività vietata sulla base della legge federale Computer Fraud and Abuse Act (CFAA)
    - la corte ha riconosciuto che il caso non costituisce una violazione del Computer Fraud Abuse Act

WEB SCRAPING: QUANDO LEGALE?

- legalità in una "zona grigia"
  - dati disponibili liberamente sul web
  - dati non personali
- esempi
  - fornire un servizio di confronto di tariffe di treni ad alta velocità
  - fornire dati a servizi per autorità che monitorano traffici illegali
- scraping e blockchain
  - reperimento di dati degli exchanger e confrontare la loro affidabilità
  - andamento delle transazioni degli exchangers
  - monitorare il prezzo delle criptomonete in real time e analizzare quali sono i fattori che influenzano le variazioni del prezzo
  - offrire dei servizi di monitoraggio per la polizia: raccogliere informazioni che aiutino ad individuare scam sulla blockchain
    - `ramsomware`
    - `dust attack`

WEB SCRAPING: ALCUNE REGOLE

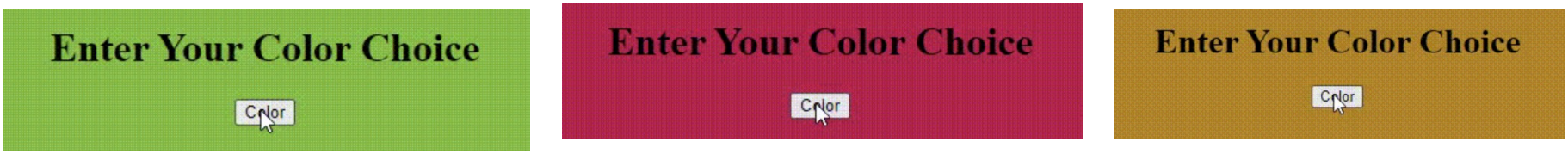
- rispettare e seguire i `Termini di servizio (ToS)` del sito web e i file `robot.txt` prima di iniziare l'attività di scraping
  - protocollo di esclusione robot (in inglese Robots Exclusion Standard) indica, le regole indicate dai gestori di un sito web ai crawler che lo visitano
  - applica alcune restrizioni di analisi sulle pagine del sito.
- effettuare richieste al sito web con una frequenza controllata.
  - evitare richieste troppo frequenti che potrebbero portare al cattivo funzionamento del sito
  - effettuare scraping in ore di minor traffico. E' più probabile essere bloccati nelle ore di maggior traffico
- evitare di raccogliere informazioni personali identificabili, dati critici o sensibili
- considerare se i dati sono protetti da copyright oppure da brevetti
- evitare di rendere pubblici i dati raccolti con lo scraping

TANTI TIPI DI SCRAPING...

- scraping di un unico sito web di un set fisso di pagine all'interno di quel sito?
  - mirato, realizzazione semplice
- scraping di un numero fisso di siti web noti?
  - scraper abbastanza mirato, necessario scrivere i codice personalizzato per ciascun sito
  - strutturare l'architettura dello scraper: parallelismo?
- crawler
  - scraping di un gran numero di siti web sconosciuti
  - nuovi pagine da analizzre aggiunte dinamicamente?
  - necessaria la rilevazione dinamica e automatica della struttura dei siti visitati

SCRAPING STATICO O DINAMICO?

- pagina web dinamica: pagina il cui contenuto è modificato a seconda dell'input dell'utente



esempio: una pagina il cui sfondo cambia ad ogni click dell'utente

  - il contenuto di una pagina web dinamica è di solito generato mediante `JavaScript`
- scraping statico:
  - non è in grado di gestire pagine dinamiche
  - scarica la pagina web e analizza esattamente il contenuto della sorgente della pagina, che non può contenere il contenuto generato dinamicamente
- strumenti per lo scraping statico in ambiente `Python`: `BeautifulSoup`, `Scrapy`
- strumenti per lo scraping dinamico in ambiente `Python`: `Scrapy`

SCRAPING : CUTTING EDGE PYTHON LIBRARIES

BeautifulSoup

- static scraping
- parsing di pagine web
- user-friendly, easy to learn

```
from bs4 import BeautifulSoup
soup = BeautifulSoup(html_doc,
                'html.parser')
for link in soup.find_all('a'):
     print(link.get('href')
```

- static scraping
- crea "spiders" per estrarre infomazioni da siti web
- non-blocking I/O calls: chiamate asincrone
- ottime prestazioni, migliori di qualsiasi altra libreria

WEB SCRAPING CON BEAUTIFUL SOAP

- il più semplice degli strumenti: sostanzialmente un parser HTML
- lo scraper può essere scritto come un semplice script Python

```
raw_html = """
<h1>Auto usate in vendita</h1>
<ul class="cars-listing">
    <li class="car-listing">
        <div class="car-title">
            Volkswagen Maggiolino
        </div>
        <div class="car-description">
            <span class="car-make">Volkswagen</span>
            <span class="car-model">Maggiolino</span>
            <span class="car-build">1973</span>
        </div>
        <div class="sales-price">
            € <span class="car-price">14.998,–</span>
        </div>
    </li>
</ul>
 """
```

```
# Analizzare il testo sorgente HTML memorizzato in raw_html
html = BeautifulSoup(raw_html, 'html.parser')
# Estrarre il contenuto del tag con classe 'car-title'
car_title = html.find(class_ = 'car-title').text.strip()
# Se si tratta di una Volkswagen Maggiolino
if (car_title == 'Volkswagen Maggiolino'):
    # Risalire dal titolo dell'auto al tag <li></li>
    html.find_parent('li')

    # Determinare il prezzo dell'auto
    car_price = html.find(class_ = 'sales-price').text.strip()

    # Visualizzare il prezzo dell'auto
    print(car_price)
```

WEB SCRAPING CON SCRAPY

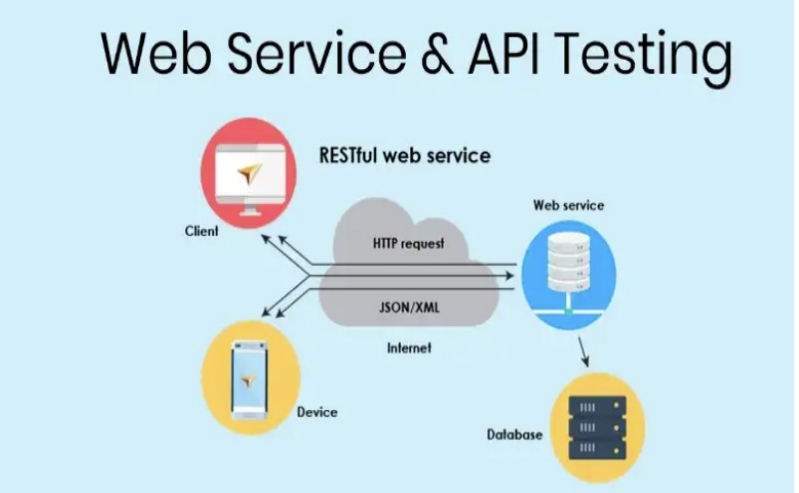
- `Python` based web crawler, open source
- scraper scritto come un programma `Python` che utilizza funzioni più avanzate rispetto a un semplice script (es: programmazione a oggetti)
- caratteristiche
  - spiders
    - data una lista di `URL` standard, accede al contenuto e lo parsa
    - crawl spider: crawling di un intero sito web
  - `CSS Selector` e parsing di espressioni `Xpath`
  - gestione della concorrenza
  - gestione dei cookies

SCRAPING DINAMICO: SELENIUM

- originariamente progettato per automatizzare il processo dei test di siti web.
- utilizza il protocollo `WebDriver` con cui interagisce e controlla un browser, installato localmente
- fornisce meccanismi per interagire con il browser e
  - cliccare bottoni
  - inserire dati nelle form
  - eseguire il codice `JavaScript`
  - analizzare il contenuto generato

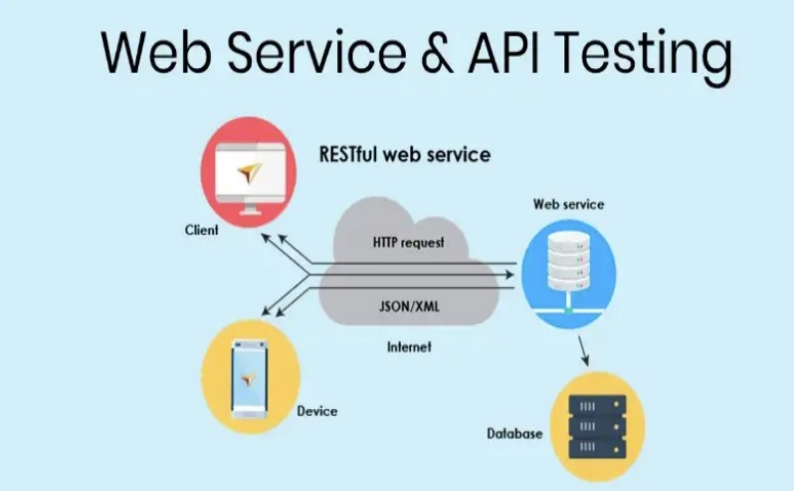
API COME ALTERNATIVA AL WEB SCRAPING

- dati accessibili mediante interfacce dette `Application Programming Interfaces (API)`
- fornite esplicitamente dal provider
  - nessun problema legale
  - limitazione del traffico implementata esplicitamente dal provider
  - forniscono dati in un formato standard (di solito `JSON`)
  - non richiesta procedura di estrazione di dati dal sorgente



API COME ALTERNATIVA AL WEB SCRAPING

- definite dalla maggior parte dei servizi
- molto spesso basate sul paradigma `REST` (`HTTP` requests,..)
- possono essere basate anche su `WebSocket`
- autenticazione tramite `API` keys
- spesso implementano misure per prevenire `Denial of Service Attacks`
  - limite al numero di chiamate API nell'unità di tempo



ANALISI DEI DATI : FASI

- pulizia dei dati
- analisi dei dati
- visualizzazione dei risultati
- tecniche diverse a seconda del tipo dei dati
  - tabellari
    - andamento del prezzo di una criptomoneta
  - grafi
    - ricercare comportamenti illeciti mediante pattern nel grafo delle transazioni di `Bitcoin`
  - …..

ANALISI: CUTTING EDGE PYTHON'S LIBRARIES

Numpy:Numerical Python

- gestione efficiente di array
- maggior efficienza rispetto a liste Python

SciPy Scientific Python

- matematica
- statistica
- si appoggia su Numpy

Pandas

- gestione e analisi di dati tabellari
- si appoggia su Numpy

PANDAS: DATA FRAMES

- gestione dei dati database-like (dati rappresentati come tabelle)
- due nuovi tipi di dato
  - `Series`
  - `DataFrames`
- caricamento di dati a partire da
  - da strutture dati `Python`
  - I/O
- ottima integrazione con `NumPy`
- ricco insieme di funzioni per il parsing di dati in formato
  - `csv`
  - `HTML`
  - `JSON`
  - `Excel`
  - `SQL`

	a	b	c
1	4	7	10
2	5	8	11
3	6	9	12

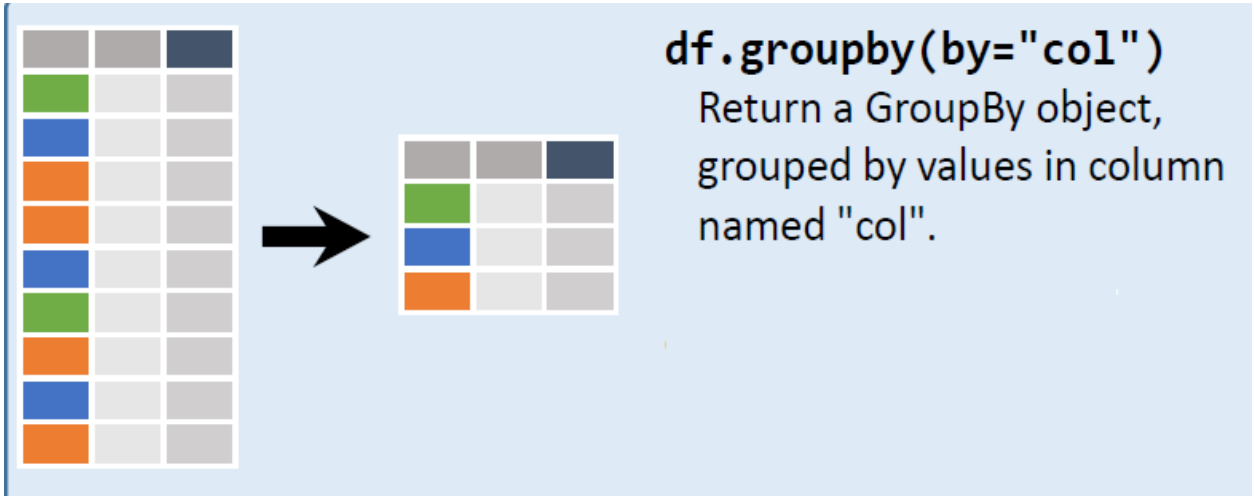
```
df = pd.DataFrame(
        {"a" : [4 ,5, 6],
         "b" : [7, 8, 9],
         "c" : [10, 11, 12]},
       index = [1, 2, 3])
```

Specify values for each column.

```
df = pd.DataFrame(
    [[4, 7, 10],
     [5, 8, 11],
     [6, 9, 12]],
    index=[1, 2, 3],
    columns=['a', 'b', 'c'])
```

Specify values for each row.

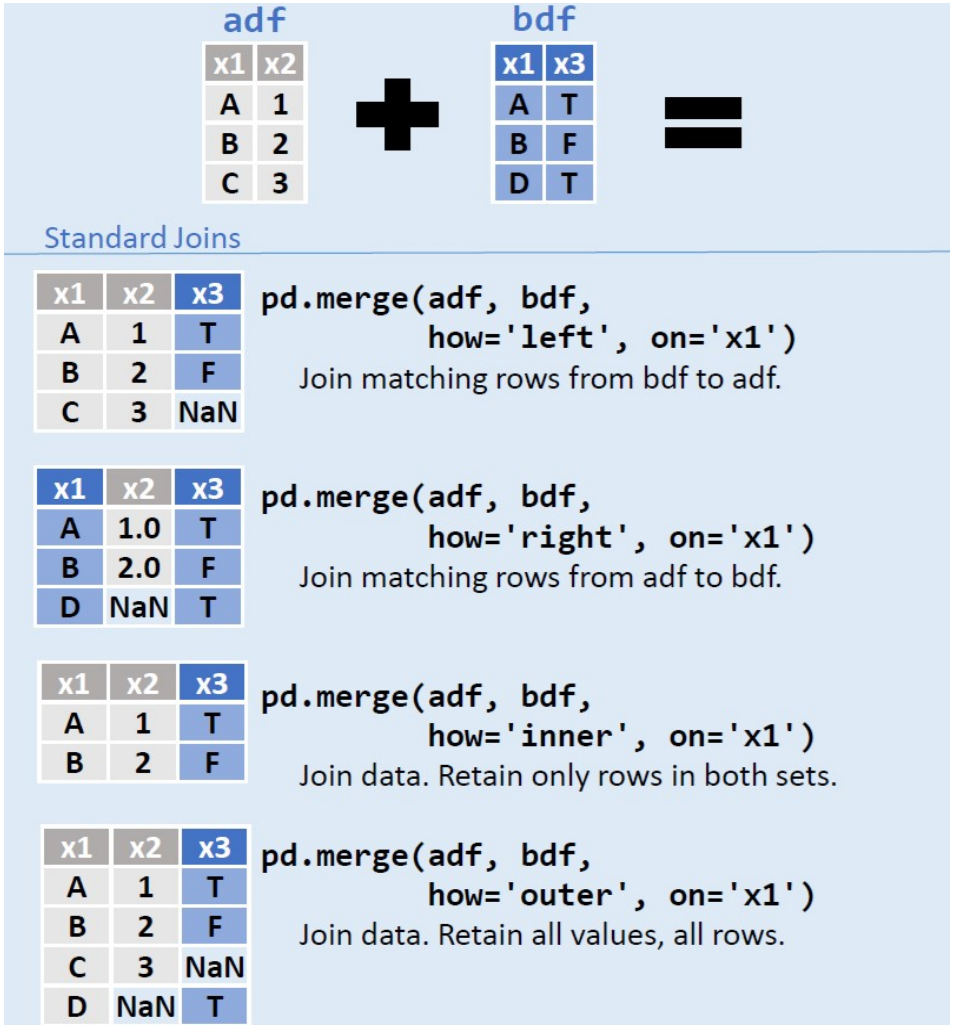
PANDAS: DATA GROUPING



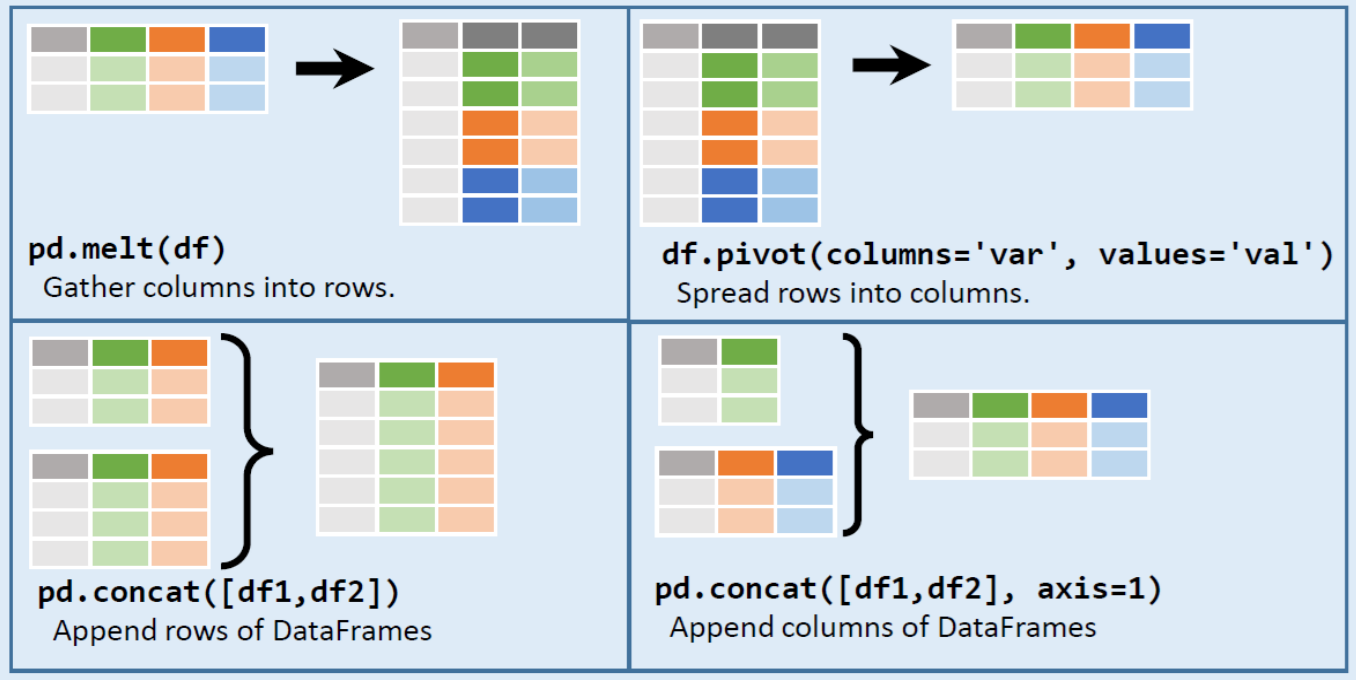
- groupby
  - raggruppare i dati per
  - applicare ad ogni gruppo una funzione di aggregazione

PANDAS: DATA COMBINING

- merge usata per il `join` tra tabelle
- comportamento simile a database standard
  - `left`
  - `right`
  - `inner`
  - `outer`



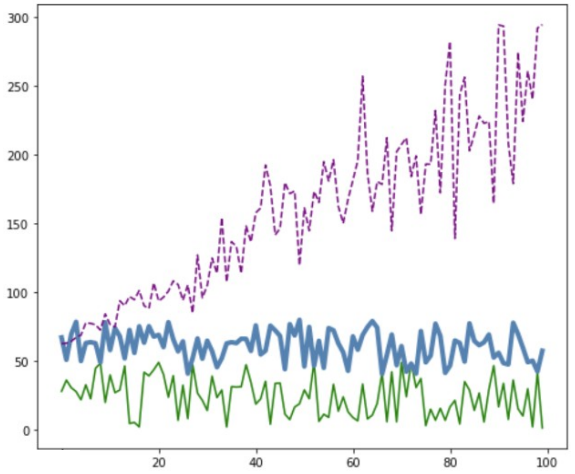
PANDAS: DATA RESHAPING



- reshaping
  - cambiare il layout dei dati
  - trasformare righe in colonne e colonne in righe
  - molte altre funzioni (sort, drop,...)

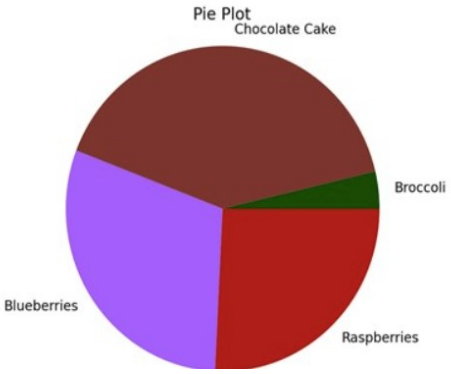
CUTTING EDGE LIBRARIES: DATA VISUALIZATION

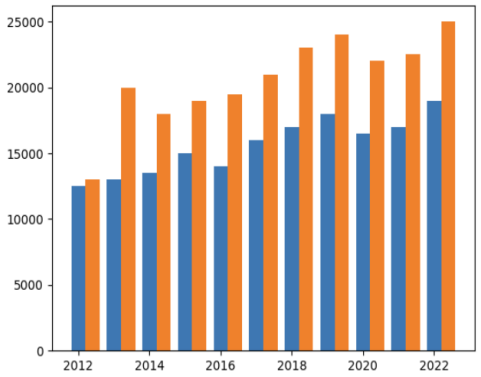
matplotlib



Matplotlib

- statistica descrittiva
- costruita su num.py
  - required dependency
- creazione di grafi 2D
- supporto limitato per grafici 3D

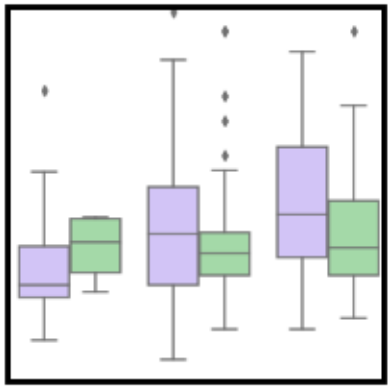




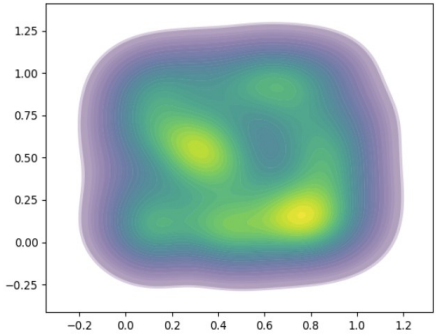
CUTTING EDGE LIBRARIES: DATA VISUALIZATION

Seaborn

- visualizzazione dati statistici
- basato su `MatPlotLib`
- maggiori funzionalità per statistica descrittiva



boxplot

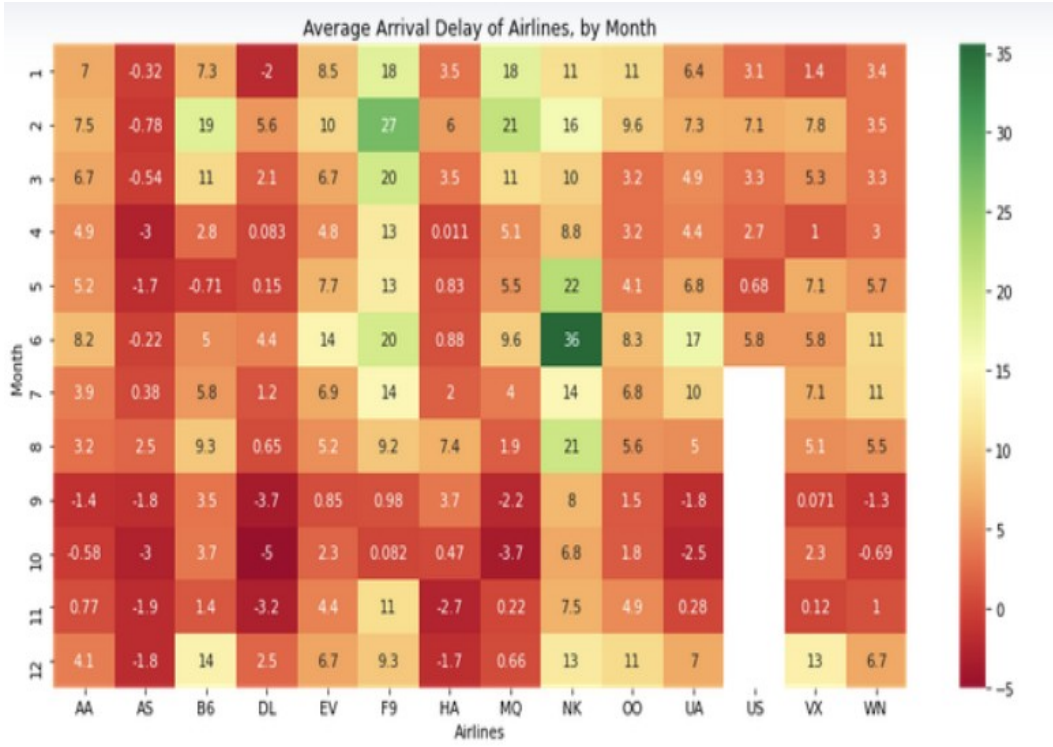


kdeplot

heatmap

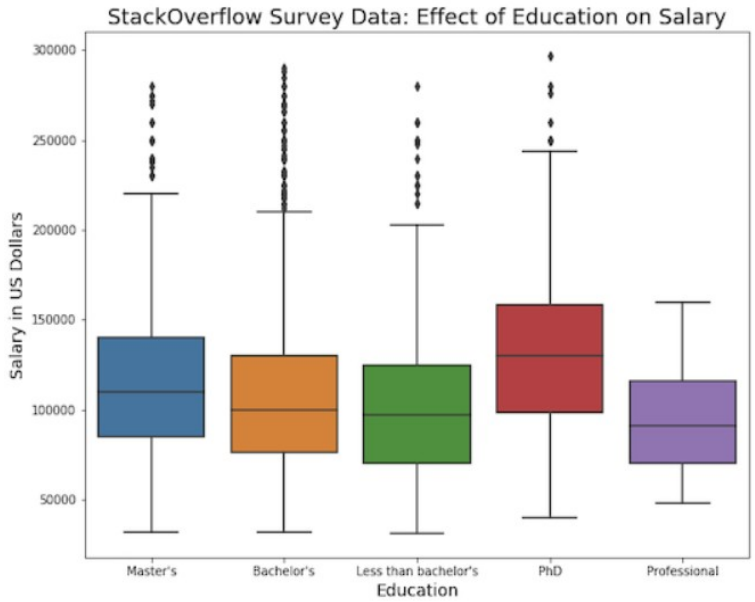
DATA VISUALIZATION

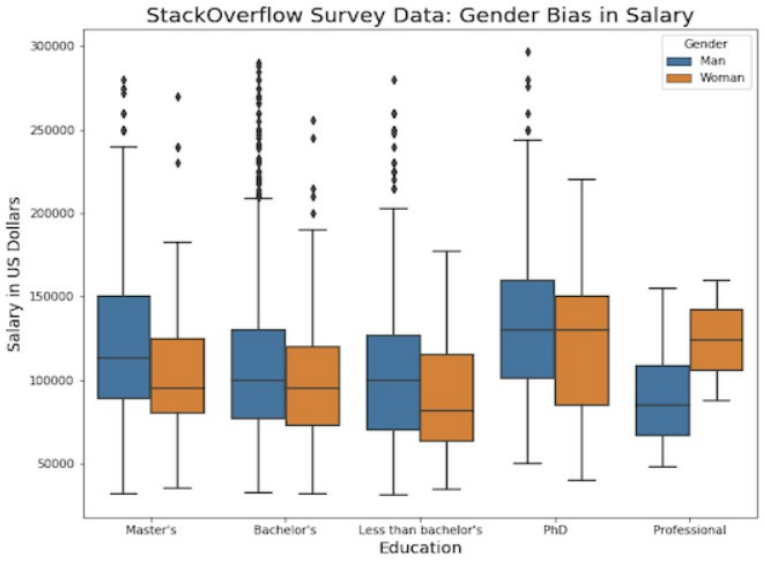
- evidenziare l'informazione nei dati tramite il grafico più opportuno
- a destra: ritardi medi di varie compagnie aeree, divisi per mese
- heatmap:
  - individuazione immediata di pattern caratteristici
  - Settembre-Novembre sono i mesi più "scuri", minori ritardi dovuti a un minor traffico in questi mesi



DATA VISUALIZATION

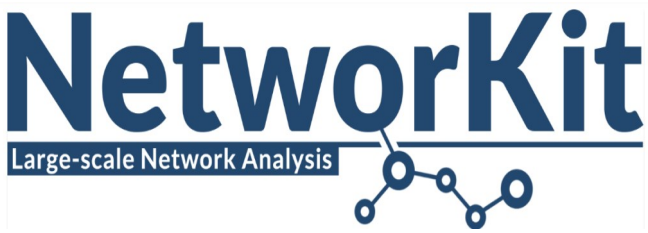
- evidenziare l'informazione nei dati tramite il grafico più opportuno
- a destra: impatto del livello di istruzione sul salario medio
- `Boxplot` visualizza
  - la distribuzione dei dati
  - quartili
  - mediana
  - minimo, massimo
  - outliers





CUTTING EDGE LIBRARIES: GRAPHS ANALYSIS

- `NetworkX`
  - analisi di grafi
  - metriche
    - diametro, gradi nodi, clustering coefficient
  - molti algoritmi standard
  - generazione di grafi con modelli noti (`random graphs`, `scale free`,..)
  - basato su `matplotlib`
  - scalabilità parziale per grafi di dimensione molto grande



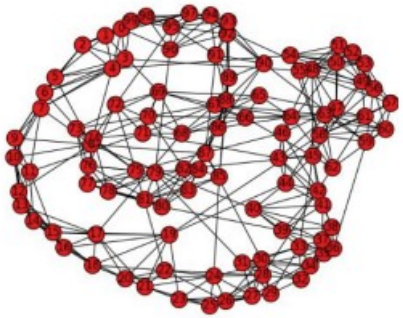
- `Networkkit`
  - algoritmi efficienti
  - parallelizzazione

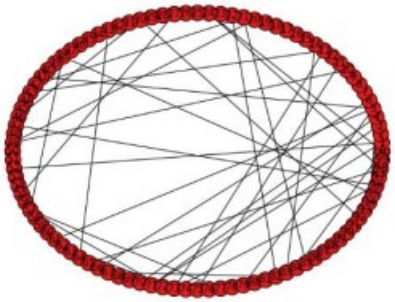
    su architetture multicore
  - stesse funzionalità

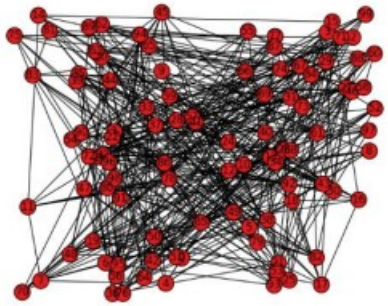
    di `NetworkX`
  - focus su scalabilità e parallelismo

LIBRARIES: ADVANCED DATA VISUALIZATION

- NetworkX
- Plotly
- Gephi
- -----





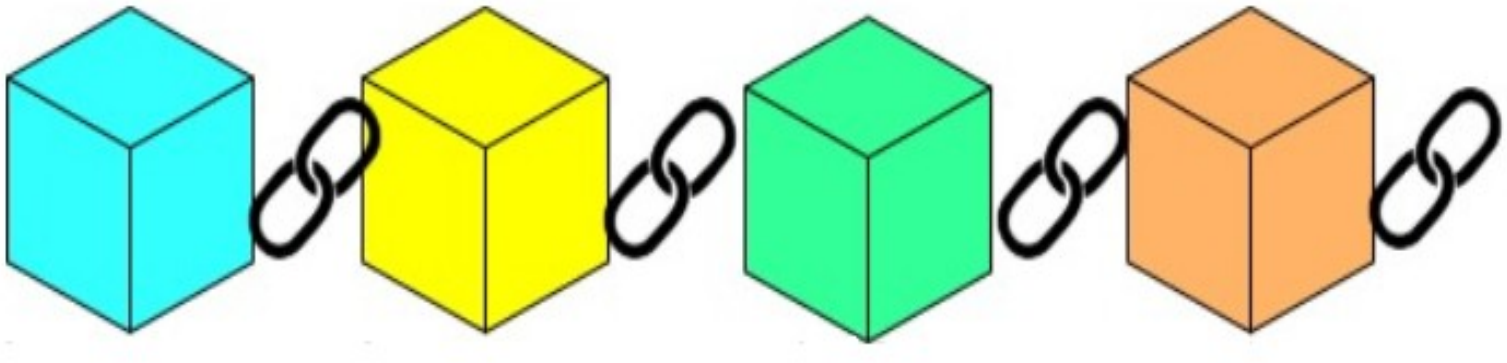


CASI D'USO: PERCHE' LE BLOCKCHAIN?

- reperire dati provenienti da blockchain
  - pubblici
  - di grandi dimensioni: un esempio concreto di "big data"
  - strutturati (blocchi, transazioni)
  - reperibili mediante diversi meccanismi (`API`, query su cloud, scraping,..)
- analizzare i dati con i seguenti obiettivi
  - identificare patterns interessanti
  - clusterizzarli
  - identificare modelli dai dati
  - visualizzarli
- perchè è utile?
  - individuare comportamenti illeciti in attori dell'ecosistema blockchain
  - individare trend di investimenti
  - analizzare comportamento sociale degli utenti

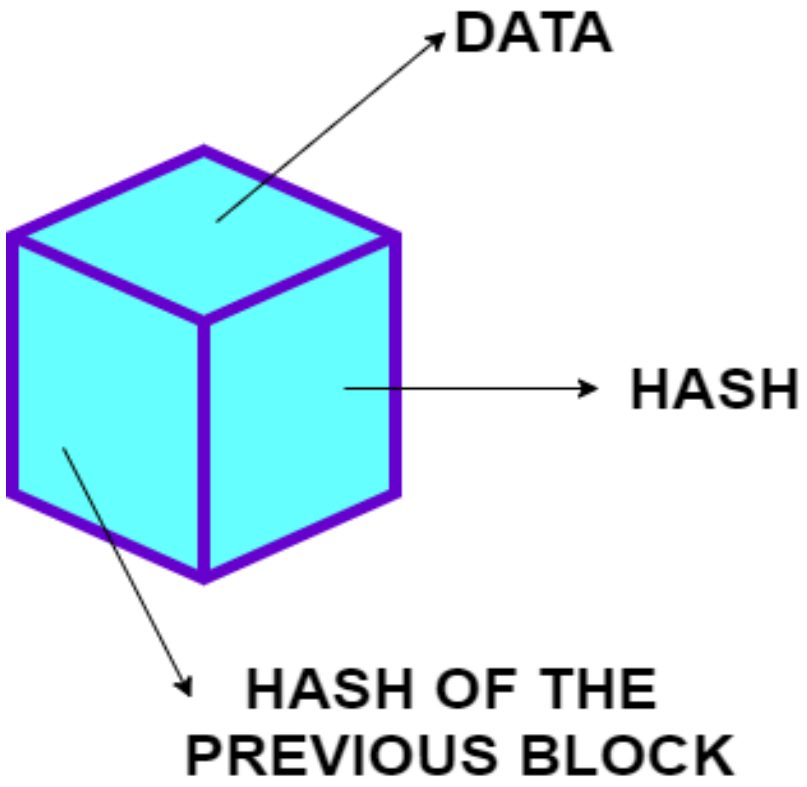
CASI D'USO: LE BLOCKCHAIN

- per motivare la scelta, diamo prima una "bird of eye view" sulla tecnologia delle blockchain



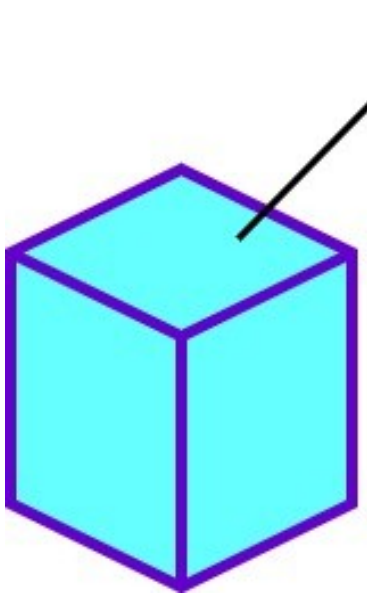
- utilizza un nuovo modello computazionale rispetto al classico client-server: peer-to-peer
- è un registro distribuito e replicato tra i nodi di una rete peer-to-peer
  - simile ad un "registro notarile"
  - caratterizzato dalla proprietà di tamper freeness

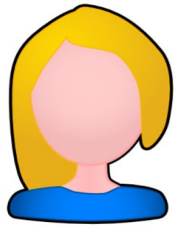
CASI D'USO: LE BLOCKCHAIN



sintesi del contenuto di un blocco della blockchain

CASI D'USO: LE BLOCKCHAIN



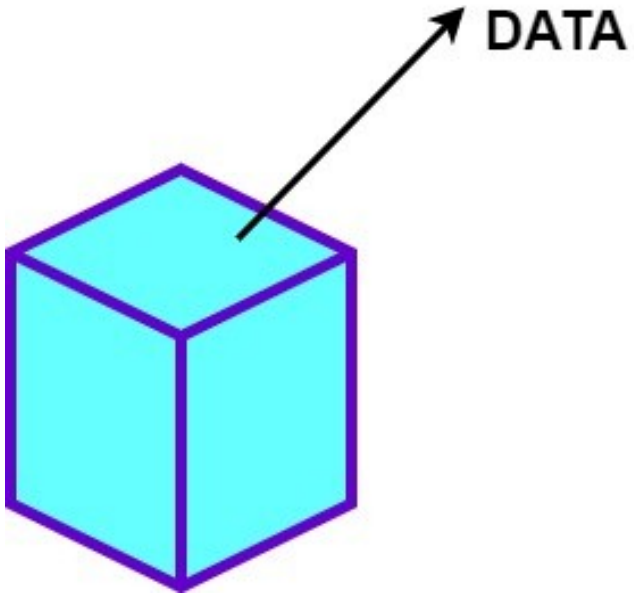


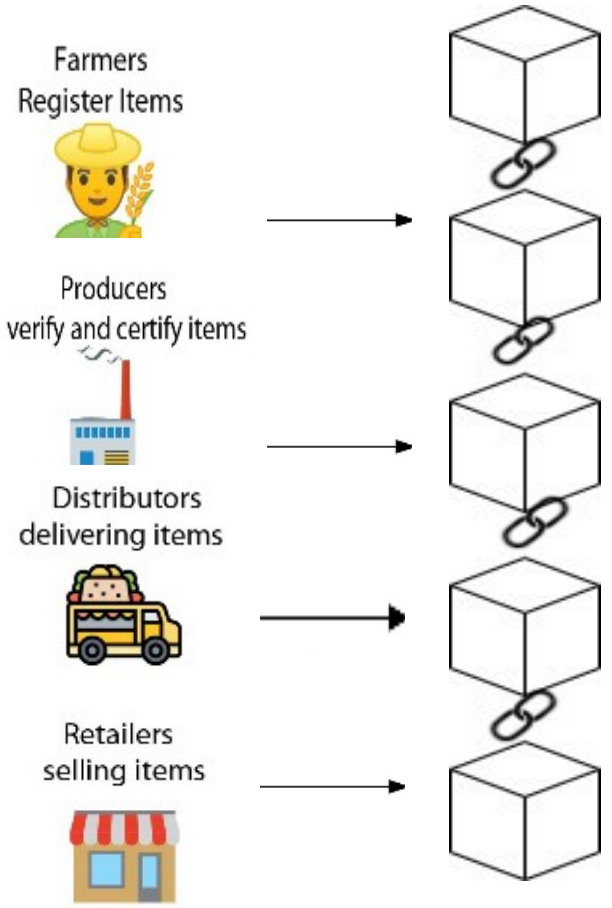
FROM

TO

AMOUNT

CASI D'USO: LE BLOCKCHAIN

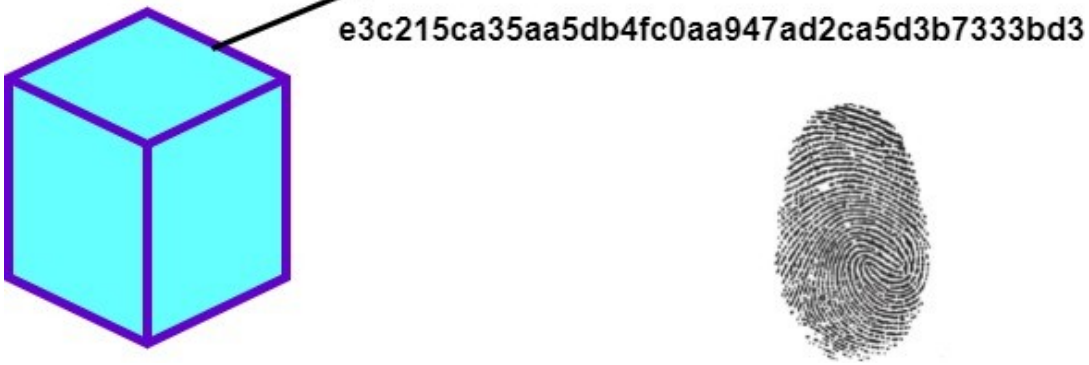


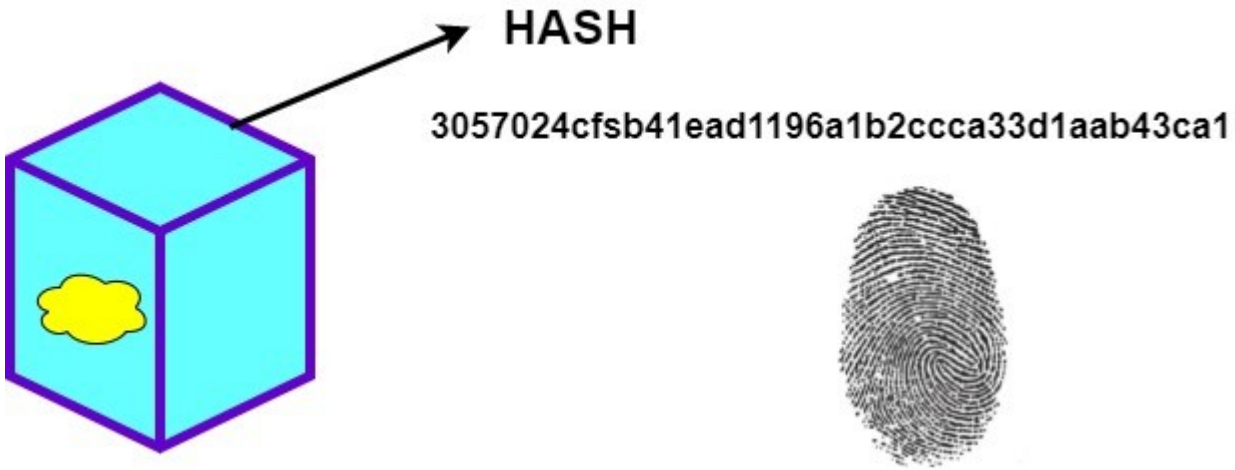


CHE DATI TOVO IN UN BLOCCO?

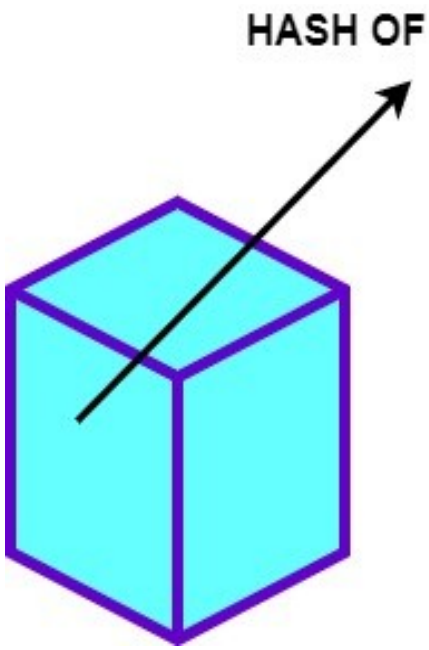
- in generale, transazioni
  - trasferimenti di cryptocurrencies (Bitcoin, Ethereum,...) o asset finanziari
- ma non solo!
  - rilevazione da sensori IoT, ECG.,...
  - tracciamento di supply chain (e.g.: diamanti,....)
  - certificazione della proprietà di un asset (NFT)
  - proprietà intellettuale/fruzione di un contenuto audio/video
  - concessione del diritto ad accedere a dati sensibili (dati medici,.....)
- in generale, processi a cui partecipano più entità indipendenti in un ambiente trustless

CHE DATI TROVO IN UN BLOCCO?



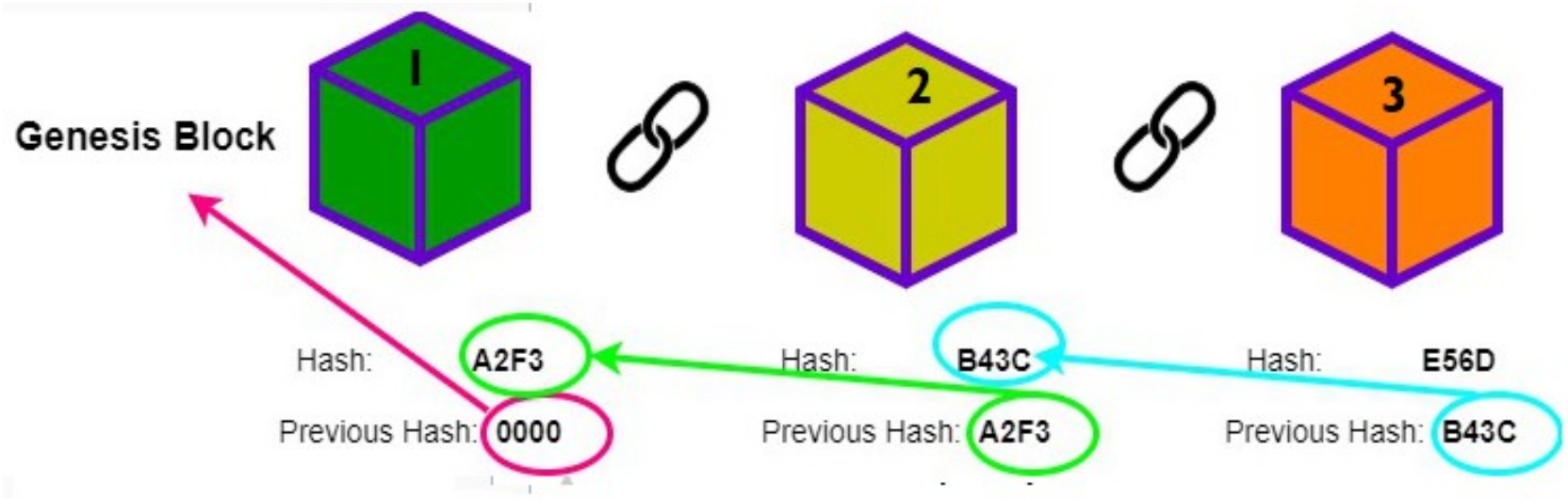


CHE DATI TROVO IN UN BLOCCO?



Hash pointers create the chain

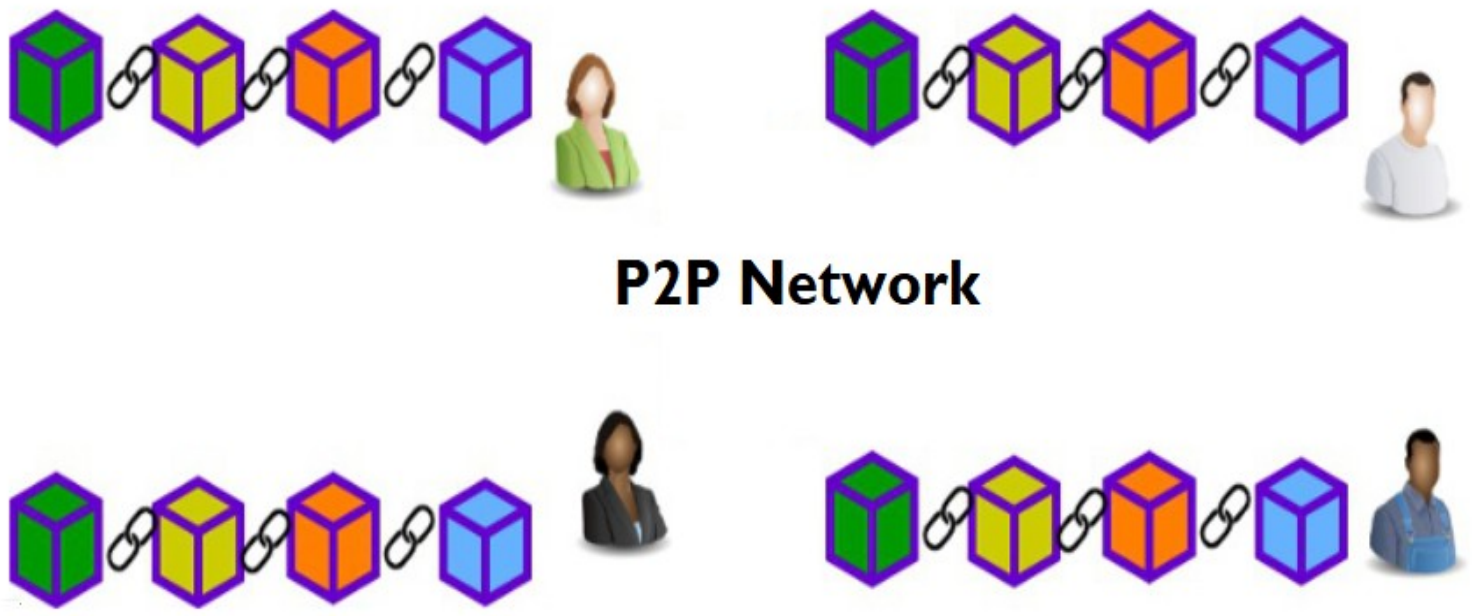
CHE DATI TROVO IN UN BLOCCO?



TAMPER FREENESS

- cambiare un hash causa la modifica di tutti i blocchi successivi
- deve essere computazionalmente complesso ricalcolare l'hash di un blocco
  - in generale molto semplice calcolare un hash
  - più difficile se il blocco contiene una informazione che è complesso calsolare
  - in blockchain difficoltà garantita da `Proof of Work`
    - altre blockchain usano meccanismi diversi

IL REGISTRO REPLICATO E DISTRIBUITO



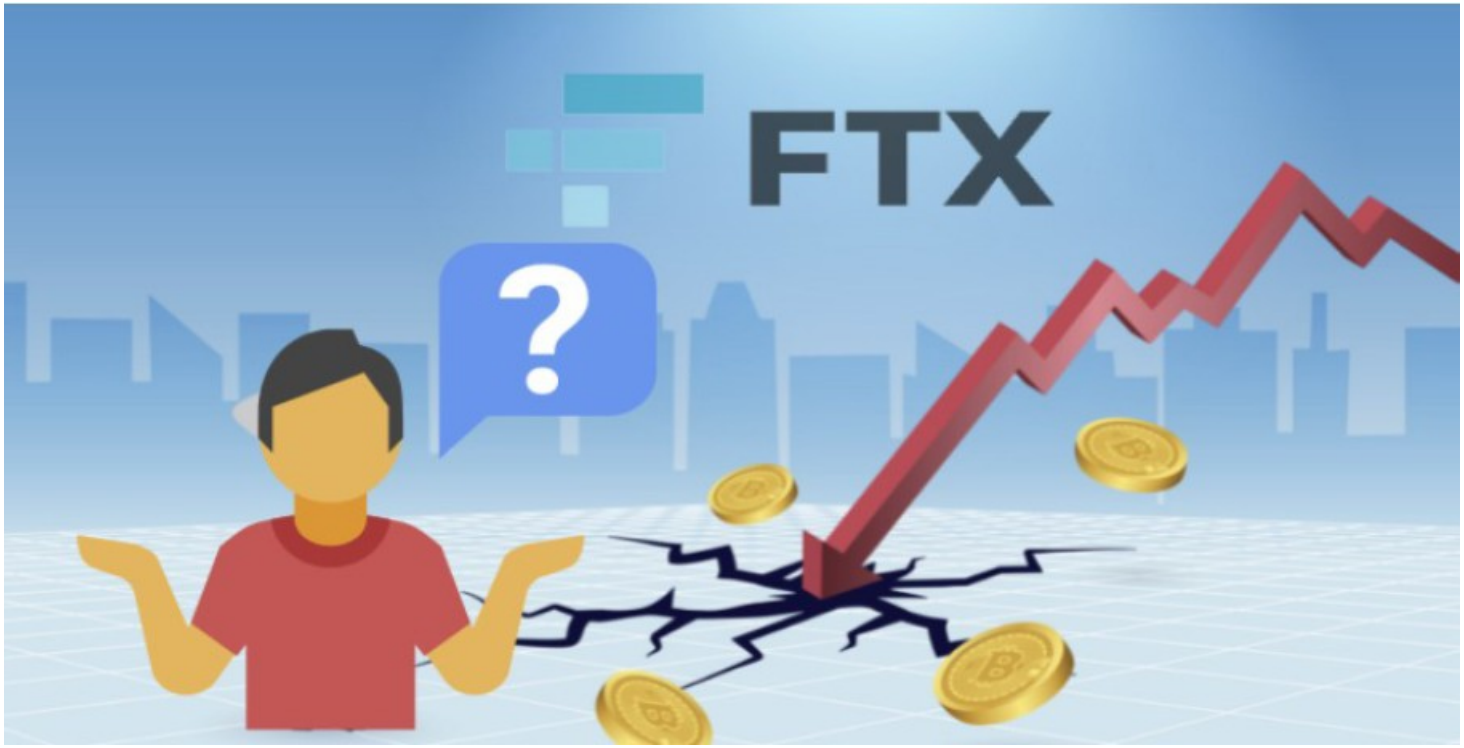
- chi decide quale blocco aggiungere alla blockchain?
  - distributed consensus

BLOCKCHAIN ANALYTICS: MOTIVAZIONI

- analisi di dati provenienti da blockchain è utile in diversi scenari
- le slide successive mostreranno alcuni risultati di analisi svolte nell'ambito del mio gruppo di ricerca

  `https://sites.google.com/unipi.it/pisadltlaboratory`
- alcuni esempi di analisi di:
  - crypto-Exchangers
  - distribuzione della ricchezza in in `STEEMIT`
  - betting pattern all'interno della blockchain di `Bitcoin`

L'ECOSISTEMA DEGLI EXCHANGERS



- FTX: uno degli exchangers più famosi e fidati dell'ecosistema crypto
- 14 November 2022: FTX bankruptcy

COME SCEGLIERE UN EXCHANGER?

kraken

?

Bitstamp

- Reputazione
- Tecnologia
- Commissioni
- Numero di asset negoziabili
- User experience
- Volume scambiato

PERCHE' IL VOLUME E' IMPORTANTE?

- Sinonimo di successo
- Più volume più liquidita
- Attrae più utenti
- Indicizzati meglio da CoinMarketCap

EXCHANGERS: ANALISI DI AFFIDABILITA'

- il volume che un exchanger riporta a `CoinMarketCup(CMK)` è reale?
- confronto del volume reale riportato da `CMK` con le transazioni reperite dalle `API` di diversi exchangers

Volume Bitstamp

Transazioni Bitstamp

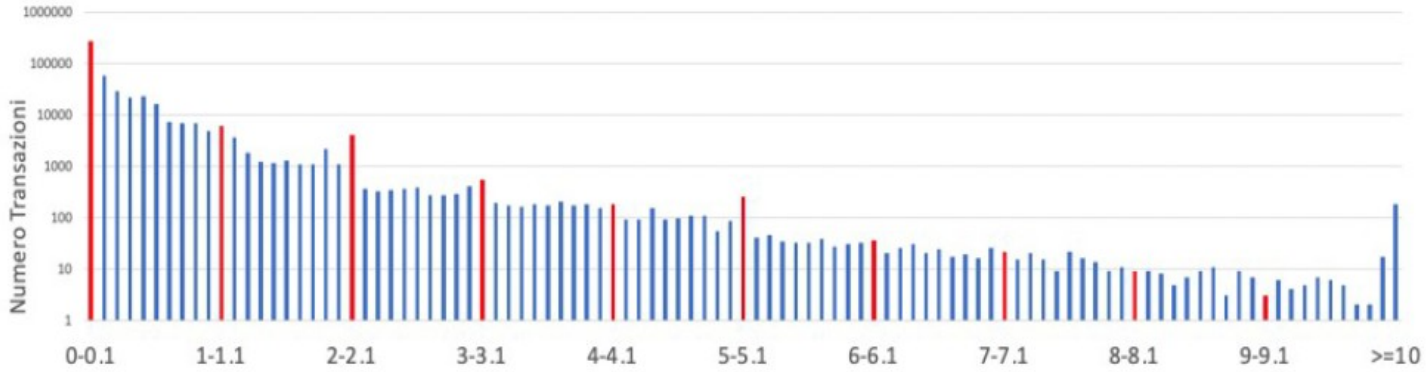
EXCHANGERS: ANALISI DI AFFIDABILITA'

- il volume riportato a `CoinMarketCup(CMK)` è reale?

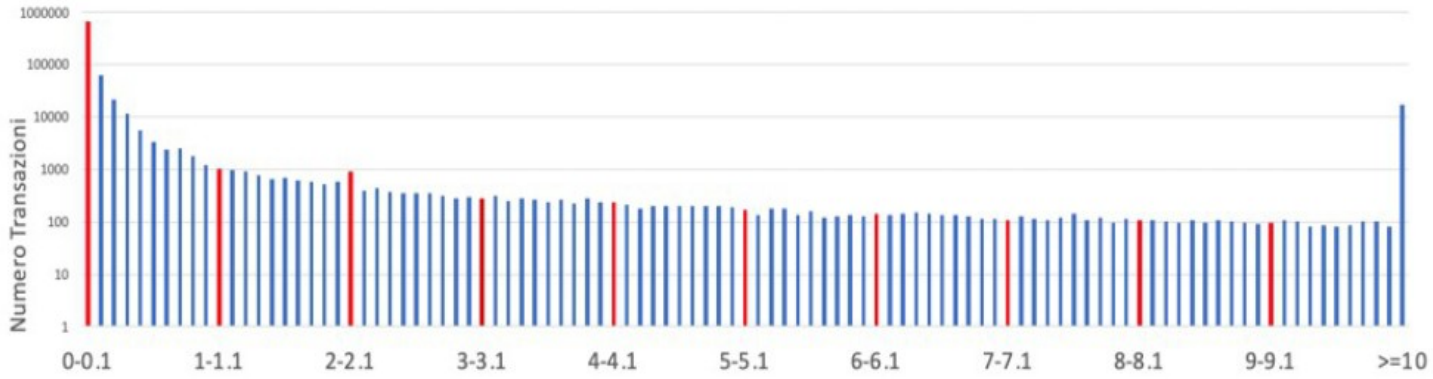
- la relazione tra volume e numero di transazioni si perde per questi exchanger

EXCHANGERS: VOLUMI SCAMBI BTC-USD

Kraken



HitBTC

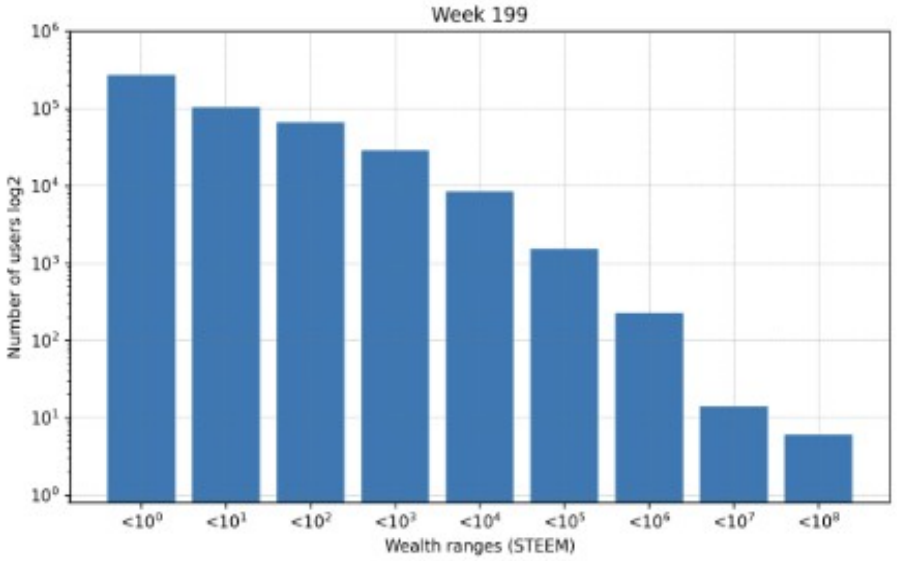


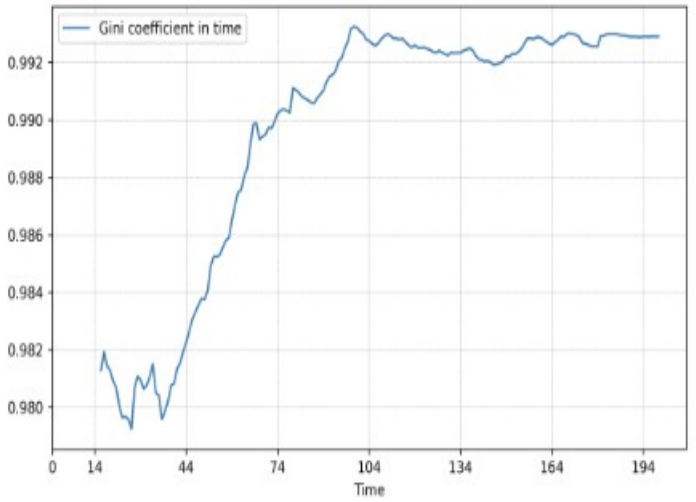
- distribuzione dei volumi di scambio BTC-USB
- HitBTC: troppo regolare per essere vero! Distribuzione generata artificialmente

DISTRIBUZIONE RICCHEZZA IN STEEMIT

- STEEMIT
  - Blockchain Online Social Media (BOSM) implementata sulla blockchain STEEM
  - public e permissionless blockchain
  - diversa da Bitcoin e Ethereum perchè disegnata per supportare applicazioni social
- ricompense distribuite nell'ecosistema di STEEMIT
  - posts: author e curation reward
  - witnesses: creation reward
- analisi della ditribuzione della ricchezza
  - come si distribuisce la ricchezza tra i vari attori?
  - fenomeno *"the rich get richer"*

DISTRIBUZIONE RICCHEZZA IN STEEMIT





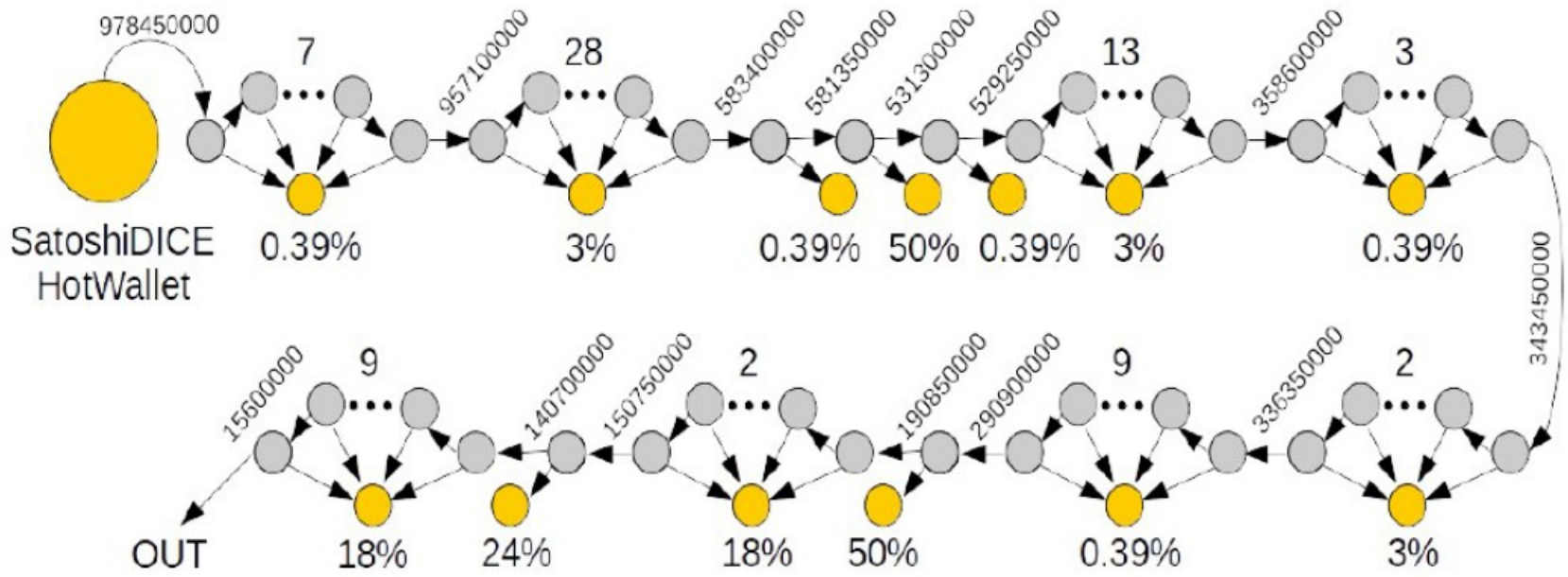
- il maggior numero di utenti ha meno di `1.000 Steem`
- meno di `1000` utenti posseggono la maggioranza della ricchezza

- coefficente di `Gini`:
  - disuguaglianza nella distribuzione di una ricchezza
  - valori maggiori, maggior disuguaglianza
  - valore molto alto per `STEEMIT`

SATOSHI DICE: ANALISI DI BETTING STRATEGIES

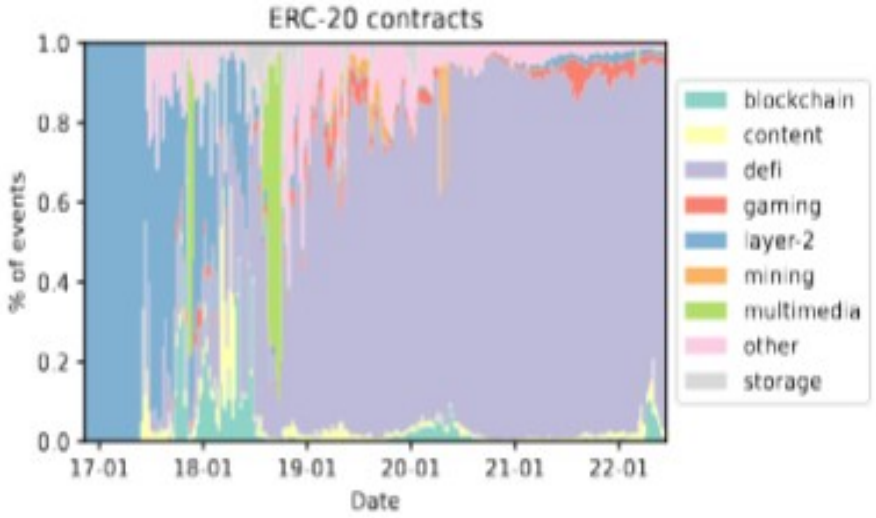
- servizio di betting attivo sulla blockchain di `Bitcoin` nel periodo `2012-2014`
- per giocare
  - inviare una transazione ad uno degli indirizzi appartenenti al servizio
  - ogni indirizzo caratterizzato da una diversa probabilità di vincita e quindi da un diverso payout
  - estrazione di un numero casuale
  - l'utente può ricevere in risposta il payout, se vince, oppure un importo molto basso, corrispondente a pochi `Satoshi`
- analisi della blockchain
  - quali strategie sono utilizzate dagli utenti per giocare?
  - quante delle scommesse sono generate automaticamente da betting services?

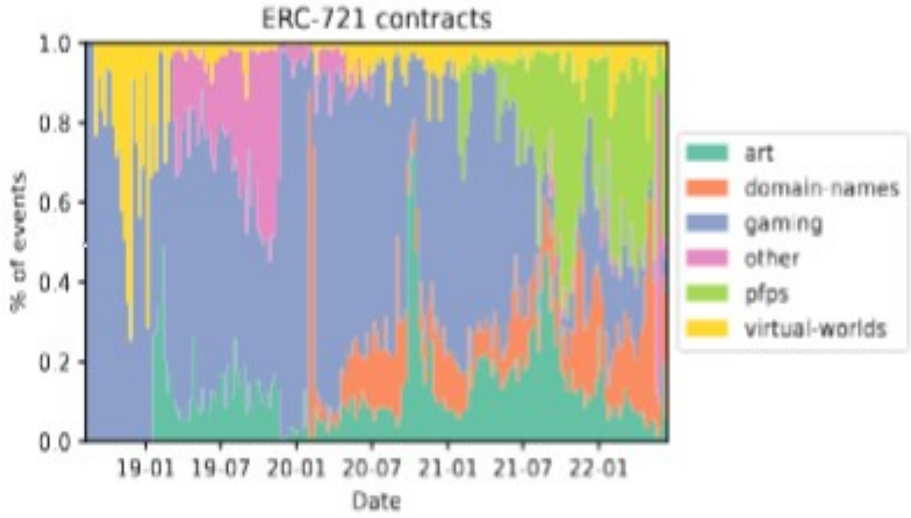
SATOSHI DICE: ANALISI DI BETTING STRATEGIES



- analisi dei betting address di `Satoshi Dice`
- lunghe catene di transazioni con pattern particolari
  - "wheel pattern"
  - caratteristici di un comportamento generato da bot
  - pattern particolari "`martingale`", "`D'Alambert`"

ANALISI DI CONTRATTI ETHEREUM ERC-20 E NFT

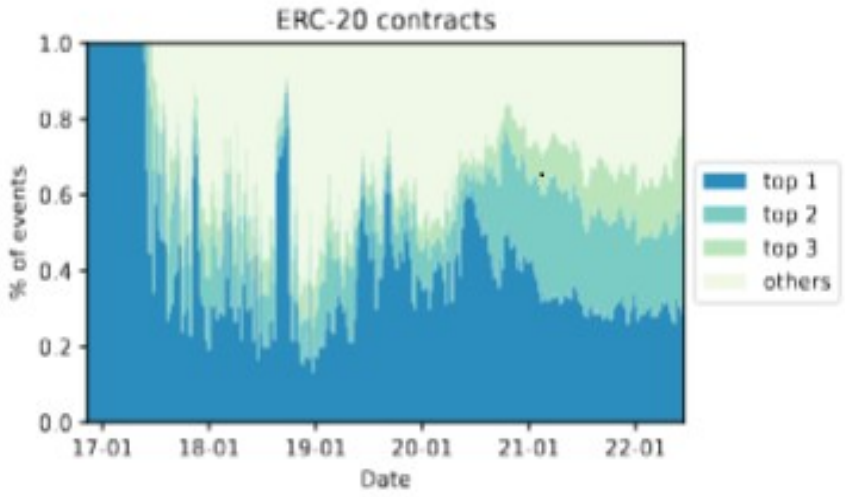


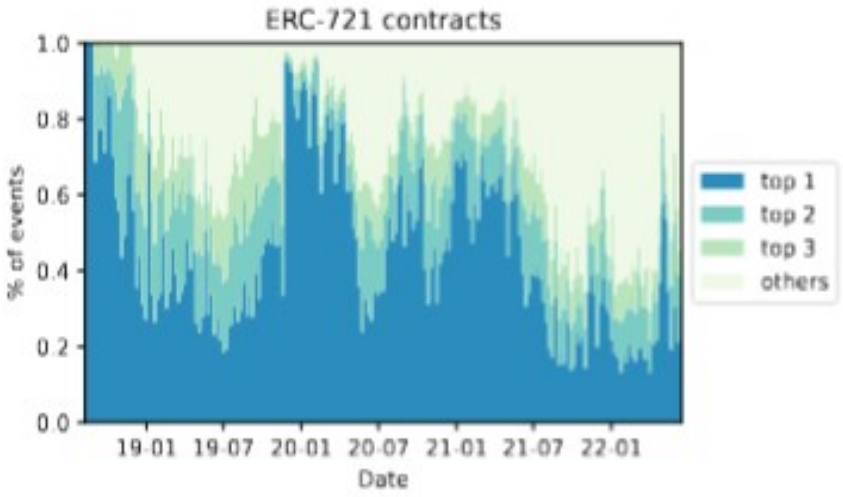


Evoluzione del tipo di contratti ERC-20 e ERC-721 tra il 2017 e il 2021

Analisi effettutata sui primi 100 contratti Ethereum in termini di trasferimenti

ANALISI DI CONTRATTI ETHEREUM ERC-20 E NFT





analisi temporale dei traferimenti effettuati dai primi 100 contratti Erc-20 e NFT

pochi contratti responsabili della maggioranza dei trasferimenti

THE PISA DISTRIBUTED LEDGER LAB

- Permanent/semi-permanent position
  - *Laura Ricci* associate professor
  - *Fabrizio Baiardi*, full professor
  - *Barbara Guidi*, RTD-B
  - *Damiano Di Francesco Maesa*, RTT
  - *Andrea Michienzi, RTD-A*
- Phds
  - *Domenico Tortola*
  - *Ricardo Lopez Almeida*
  - *Andrea Pelosi*
  - *Francesco Donini*
  - *Giuseppe Galano*
  - *Yitbarek Yimame*
  - *Calogero Turco*
- *Post Doc*
  - *Matteo Loporchio*
- Collaborations
  - *Paolo Mori, IIT CNR, Pisa*
  - *Anna Bernasconi,* University of Pisa
  - *Andrea De Salve, ISASI, CNR, Lecce*
  - *Roberto Di Pietro*, Hamad Bin Kalifa University, Quatar
  - *Nishanth Sastry*, University of Surrey

https://sites.google.com/unipi.it/pisadltlaboratory

e-mail: laura.ricci@unipi.it

CONCLUSIONI

also Harry Potter can code Python!