Deloitte.

Professional Growth Platform

by Deloitte

Deloitte Data Science Academy

Pavel Milička

MAKING AN IMPACT THAT MATTERS since 1845

Other events organized

26.-27.10. 2023	**SAFe® 5 Product Owner / Product Manager**	**workshop**
1.11.2023	**Data Storytelling**	**workshop**
9-10.11.2023	**Leading SAFe**	**workshop**
16.11.2023	**OKR (Objectives and Key Results)**	**workshop**
22.11.2023	**Datová gramotnost pro začátečníky**	**workshop**
23.-24.11.2023	**SAFe® 5 Scrum Master (SSM)**	**workshop**
30.11. 2023	**Data-driven HR**	**workshop**

More info at akce.deloitte.cz

Expectations?

- What are your expectations?
- What do you want to learn, why did you choose this course?
- What is your background, how familiar are you with SQL, Programming, Data Science?
- What tools do you use on daily basis?

Goal

Move from zero level to intermediate level in data science.

- Learn basics of Python
- Use Python and its data science tools
- Understand standard machine learning tasks
- Have an overview of methods and use cases related to AI
- Develop models for Clustering, Classification and Regression
- Be able to evaluate models

Deloitte
Data Science
Academy

Lecturers' introduction

Pavel Milička

CE AI Lead

- Master degree in Artificial Intelligence from Czech Technical University, co-author of 3 research papers – Bio-inspired hexapod control, Bilevel optimization.

- Model for predicting performance based on CVs.
- Expedition optimization for major steel manufacturer.
- Technical owner of Deloitte Dynamic Pricing solution.
- Leading project for Debt Collection Optimization for foreign Healt Insurer.
- Collaborating on automated open source intelligence system.
- Creator of internal Task Mining solution from activity logs.
- Invoice OCR Solution.
- Leading team of approx. 30 AI Specialists across CE.

Content

1	Introduction to Data Science, Data Science Tools, Python as a programming language, Python Basics, Git – code collaboration, Pure Python Data Exploration	HA1
2	Features and, Clustering, behavioral segmentation, Hands on training for segmentation with Python	HA2
3	Propensity models, single factor analysis, binning, binary classifiers, Hands on training for data exploration, classification and regression	HA3
4	Introduction to NLP and Computer Vision, Selected topics	HA4

Format

Overview

4 **Days**
During which we go through Data Science, Python, NLP, and much more

6 **Hours**
Every lecture has 6 hours.

4 **Home assignments**
You need **all 4 home assignments accepted to qualify for a final certificate.**

Calls and emails outside the room, PC for hands-on exercises only.

Major theme

During the course, we will work on various Data Science problems in the hands-ons or your homework. For the homework it is possible to come up with your own topics and assignments if it will have reasonable complexity. The homeworks topics are:

1) Python – coding exercise (scraping)

2) Clustering/Segmentation

3) Classification/Regression

4) Natural Language Processing / Computer Vision

You will learn how to work it end to end from data crunching, calculating predictors, target definition and feature engineering to modeling and final visualizations.

It is possible to come up with your own problems if the scale of the problem is comparable.

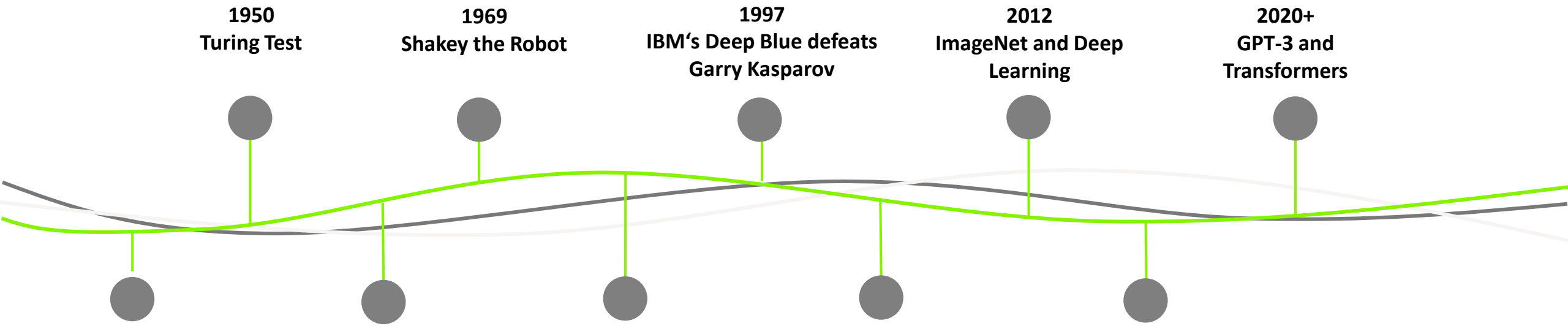
Today's program

1	**Introduction to Data Science, Data Science Tools, Keboola introduction, Python basics, Git – Code collaboration, Pure Python data exploration**	**HA1**
2	Features and dataset preparation, Clustering, Behavioral segmentation, Hands on training for data exploration and clustering with Python	HA2
3	Regression and Classification, Hands on training for data exploration, classification and regression	HA3
4	Introduction to NLP and Computer Vision, AutoML	HA4

AI/ML/DS

Where can you encounter AI?

We are living in the era of AI Revolution.



1943
Turing's Enigma, Beginnings of AI

1950
Turing Test

1956
Darthmout Workshop Establishment of AI

1969
Shakey the Robot

1980s
Expert Systems development.

1997
IBM's Deep Blue defeats Garry Kasparov

2011
IBM Watson's Jeopardy! Win

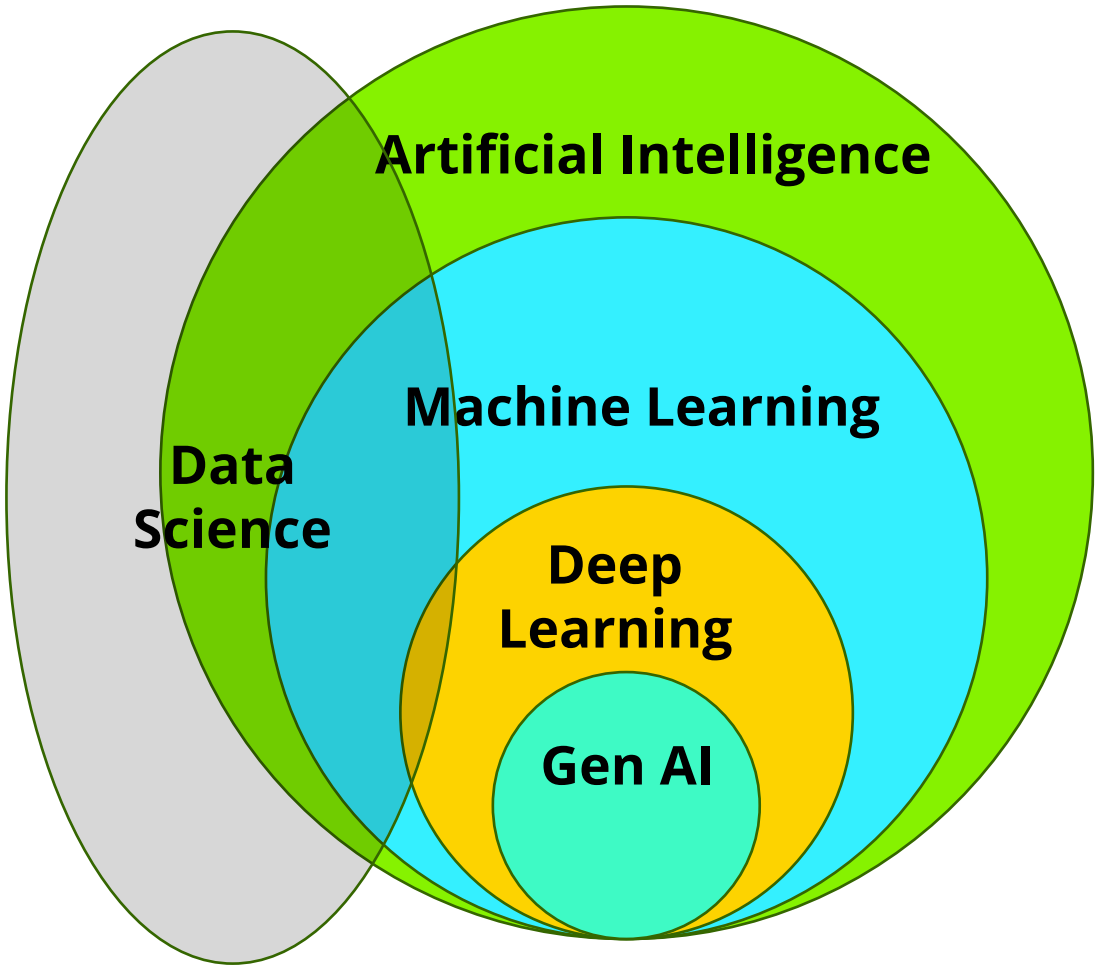
2012
ImageNet and Deep Learning

2016
AlphaGo Victory

2020+
GPT-3 and Transformers

The amount of development in the field is unprecedented in the last years.

What is AI



Artificial Intelligence is broadly defined as the ability of machines to mimic human behavior.

Machine Learning algorithms leverage statistical techniques to automatically detect patterns and make predictions or decisions based on historical data that they are trained on.

Deep learning is defined as a machine learning technique that uses deep neural networks which are able to teach computer to learn from the data that is inspired by humans.

Generative AI focuses on creating models capable of generating new content that resemble existing data.

https://synoptek.com/insights/it-blogs/data-insights/ai-ml-dl-and-generative-ai-face-off-a-comparative-analysis/#:~:text=Generative%20AI%2C%20a%20branch%20of,might%20be%20created%20by%20humans.

What is Machine Learning

Supervised Learning

„I have historical data and want to predict events/values based on them, i.e., find the dependencies and utilize them in my decision making.

- *Churn prediction*
- *Campaign targeting*
- *Demand forcecasting*

Unsupervised Learning

„I have data and want to group them together based on their characteristics."

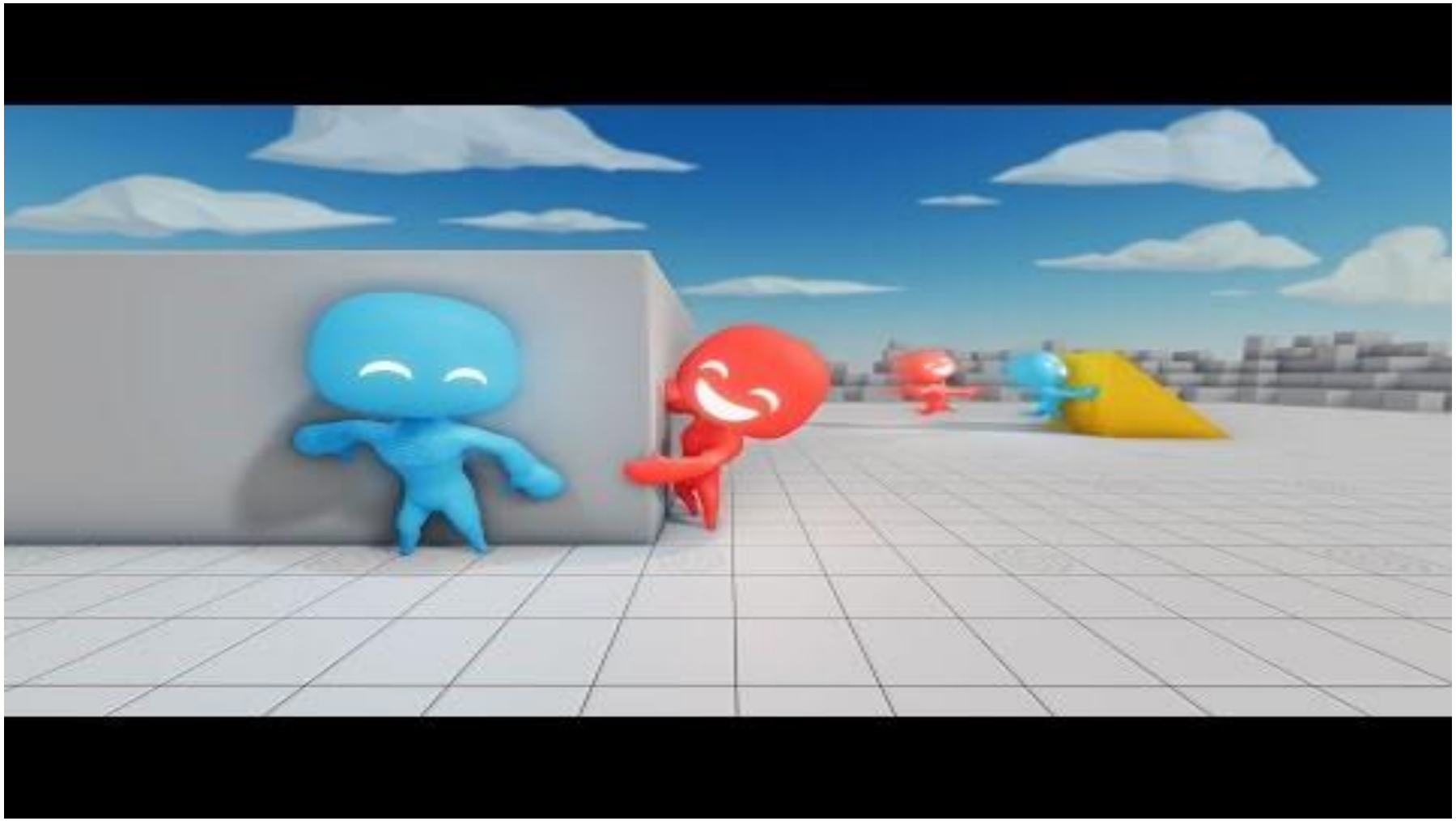
- *Client Segmentation*
- *Recommender Systems*
- *Anomaly Detection*
- *Customer Lifetime Value*

Reinforcement Learning

„I have an agent and representation of a world and I want my agent to learn how to achieve his goals by interacting with the goal."

- *Route planning*
- *Self-driving cars*
- *Gaming*

Reinforcement learning



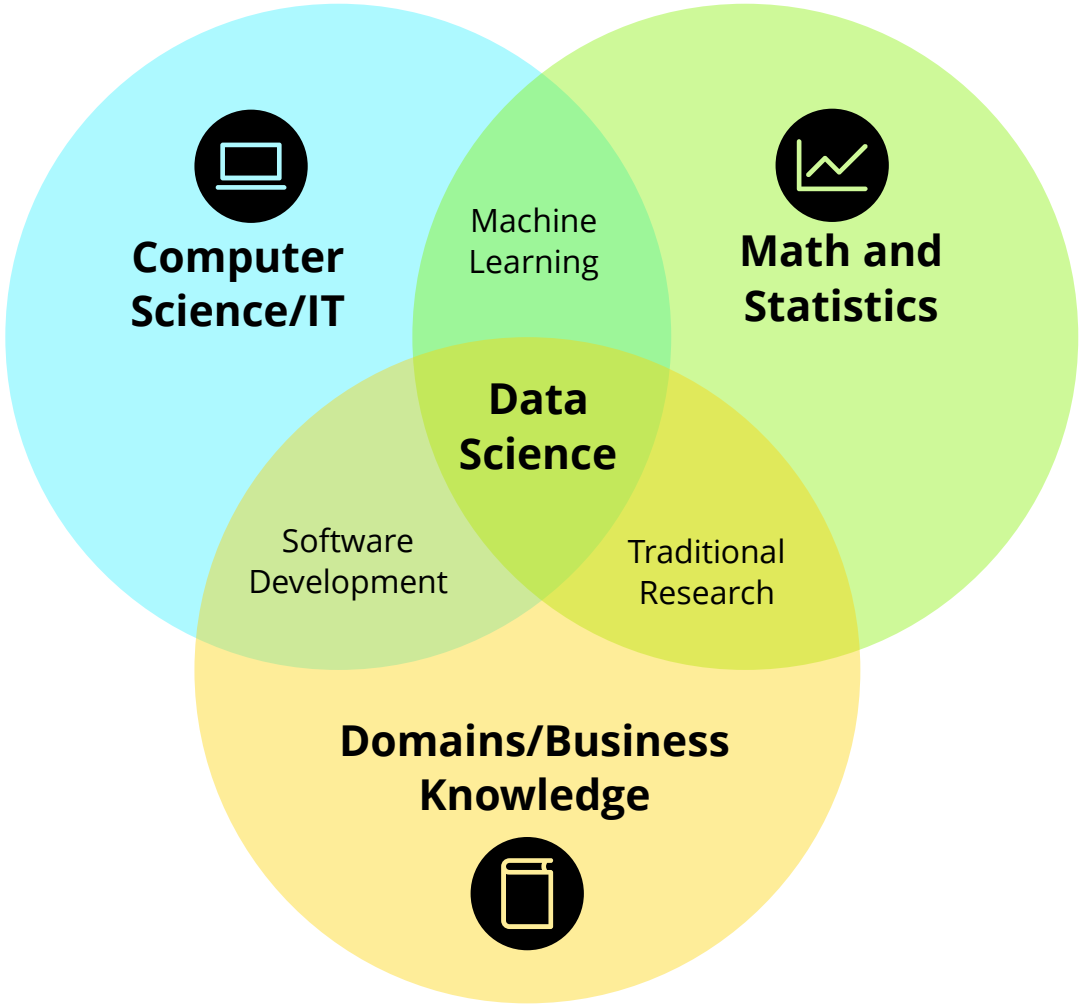
https://openai.com/blog/emergent-tool-use/

What is Data Science

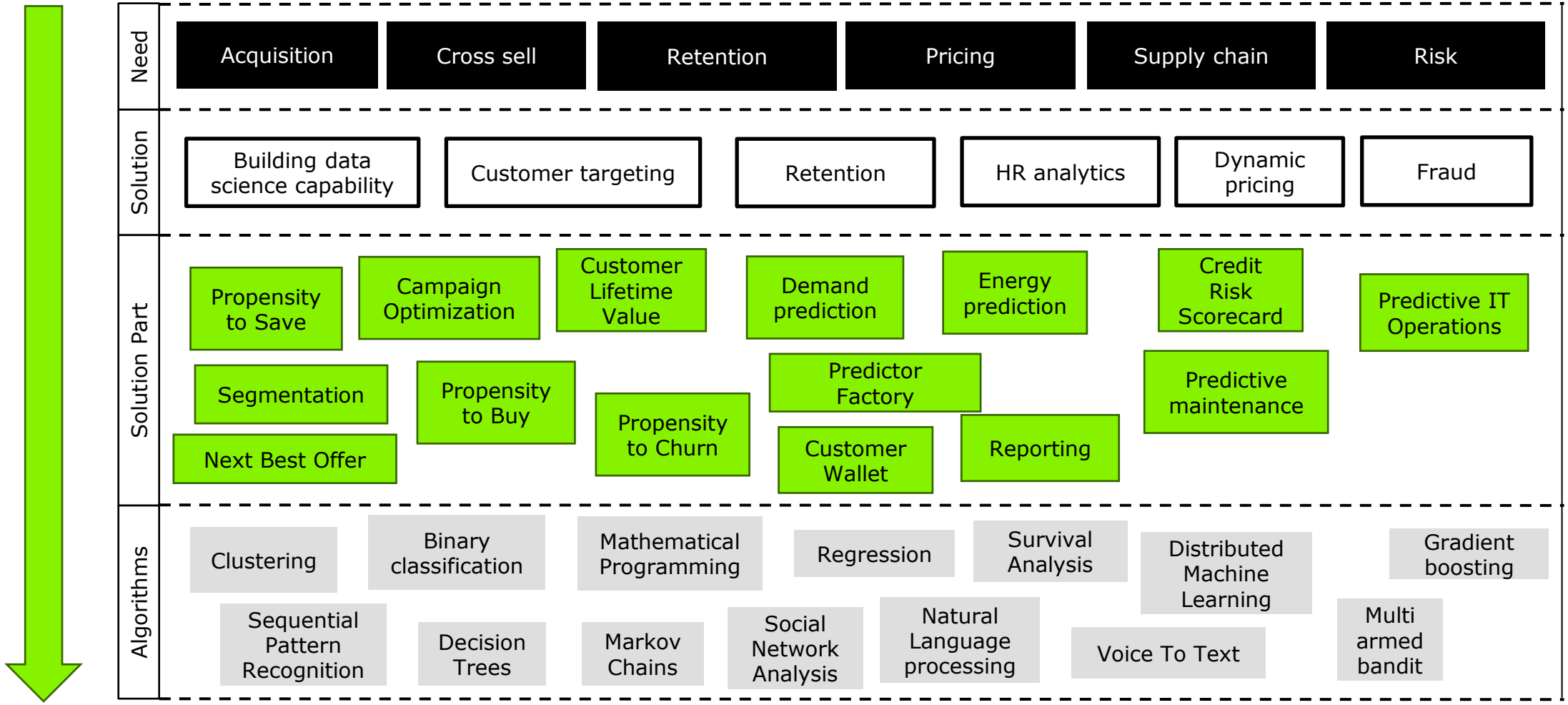
Data science is the extensive use of **data**, statistical and quantitative **analysis**, exploratory, predictive models, and fact based management to drive decisions and **actions**.

Data scientists come with flavours.

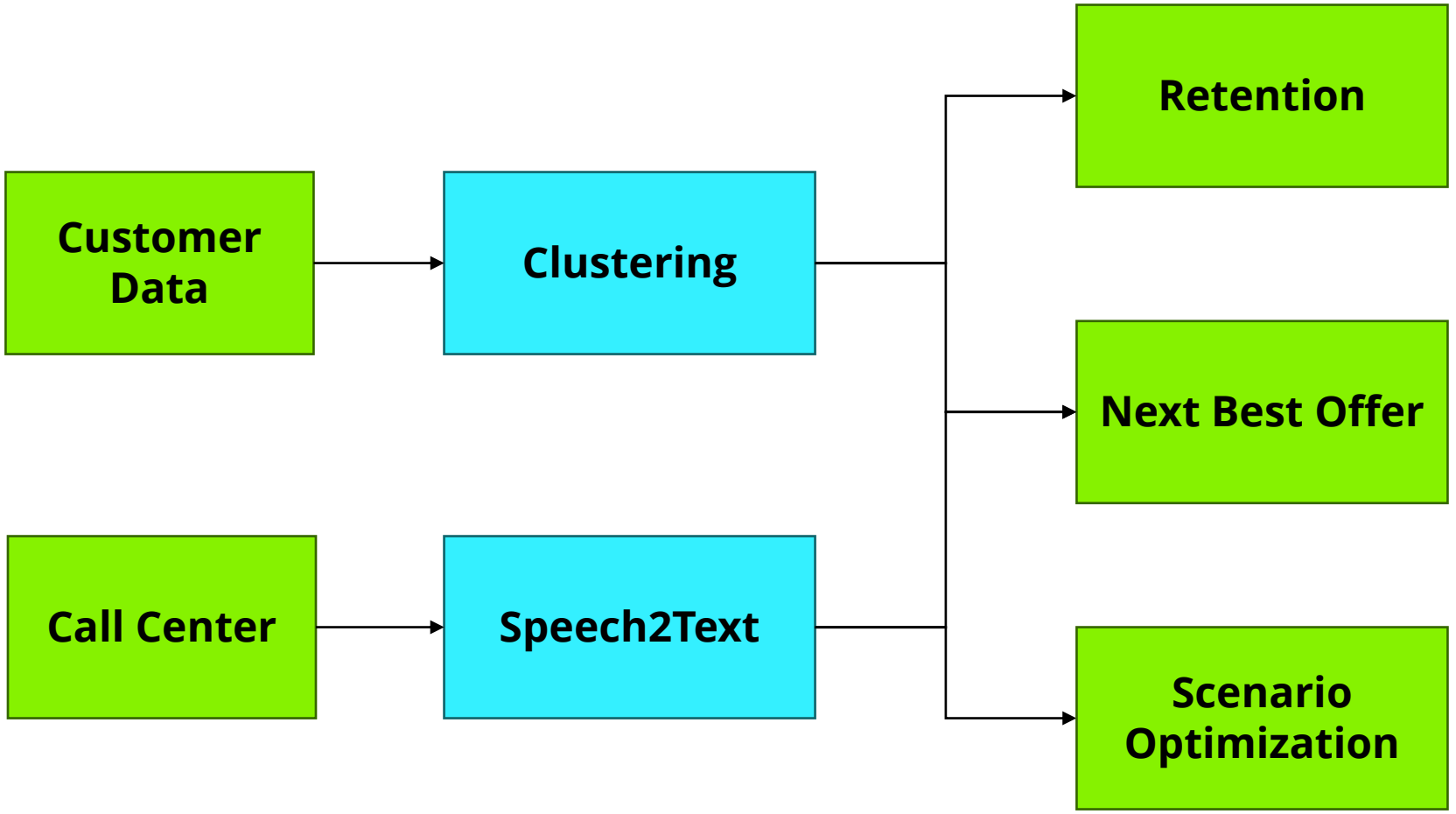
- Data Scientist
- ML Engineer
- Business Analyst
- „Prompt Engineer“



How to think about use cases



How to think about use cases – Example

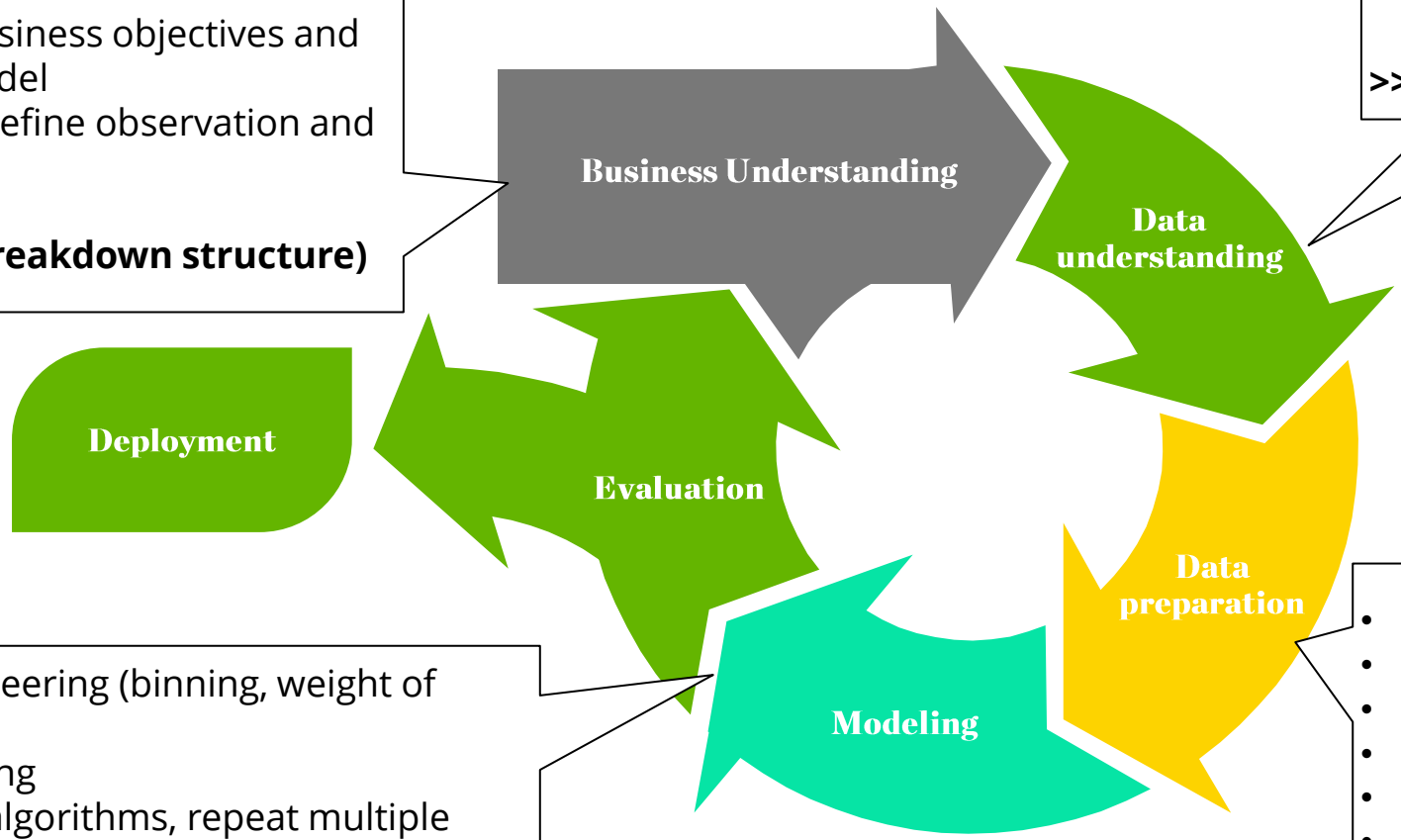


CRISP-DM Framework

Applicable to all Data Science jobs

- Define the business objectives and operating model
- Define KPIs, define observation and target
- Planning

>> WBS (Work breakdown structure)



- Discuss data availability, volumes

>> Data request

- Data ingestion
- Data quality check
- Data acceptation
- Calculate target table
- Predictors design
- Predictors calculation

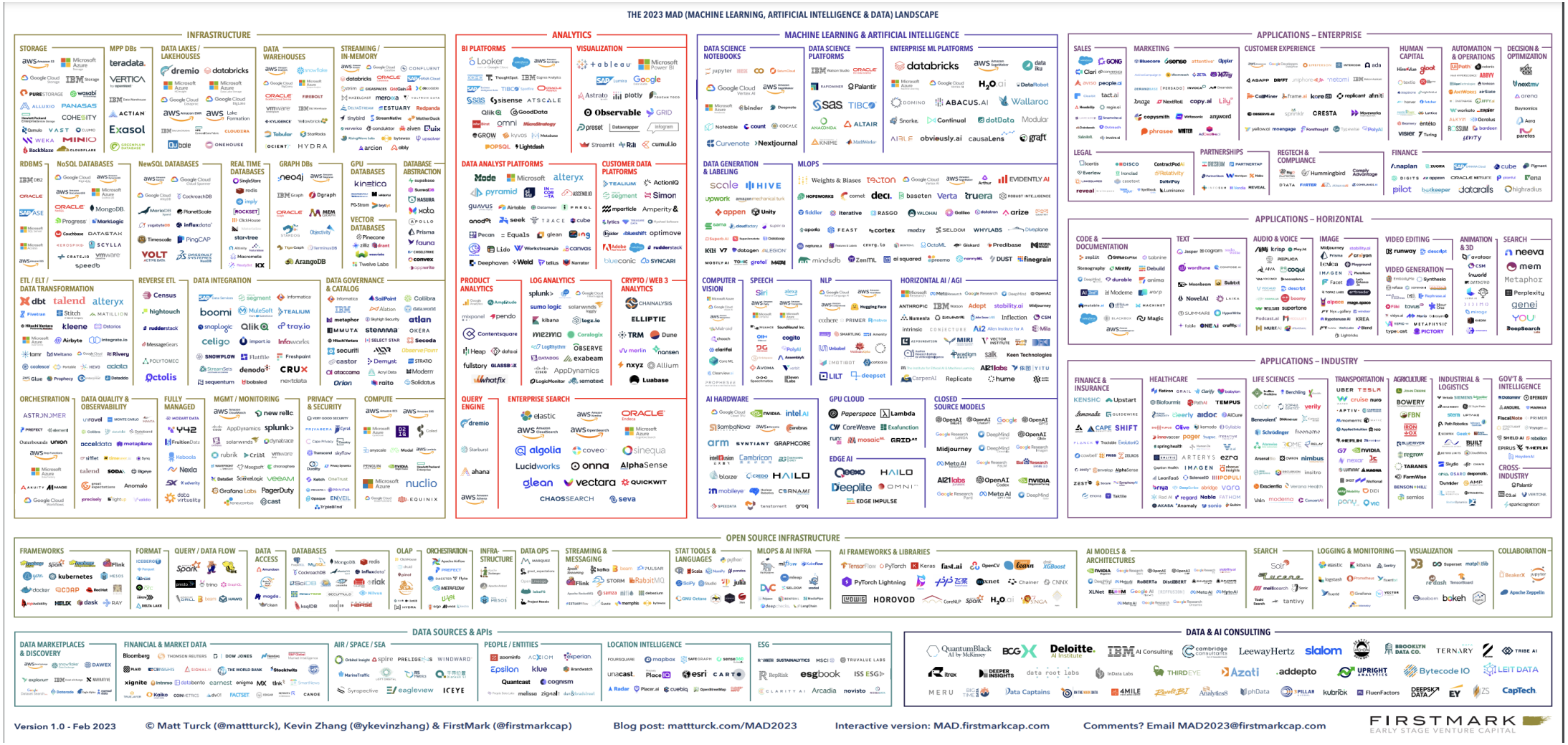
>> Modeling dataset

- Feature engineering (binning, weight of evidence, ...)
- Training, testing
- Try different algorithms, repeat multiple times

>> Model development report

Tools

AI Tools Landscape



https://venturebeat.com/ai/2023-data-ml-and-ai-landscape-chatgpt-generative-ai-and-more/

AI Tools Landscape – The Essentials

Languages and Utils
- Python
- PySpark
- SQL
- Git

Vizualize
- Python
  - Matplotlib
  - Seaborn
  - Plotly
  - Streamlit
- PowerBI
- Tableau
- Qlik

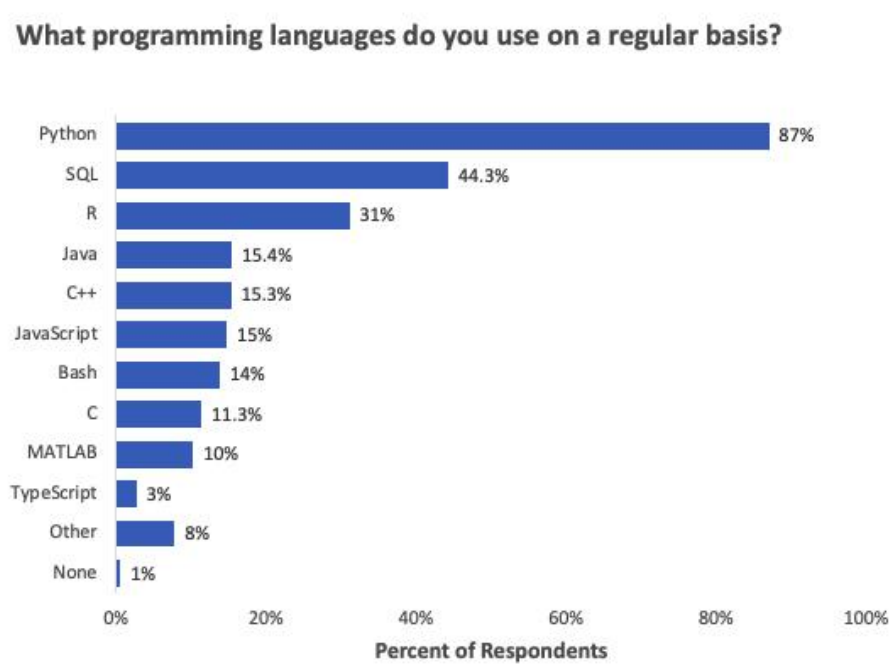
Transform
- Python
  - Numpy
  - Pandas
- SQL

Model
- Python
  - Scikit-learn
  - Tensorflow
  - PyTorch
  - PyCaret
  - ExplainerDashboard
- (OpenAI or similar GenAI tool).

Build (ML Engineering)
- Docker
- Kubernetes
- AWS/GCP/Azure
- Databricks
- Terraform
- API
- Python
  - Dagster

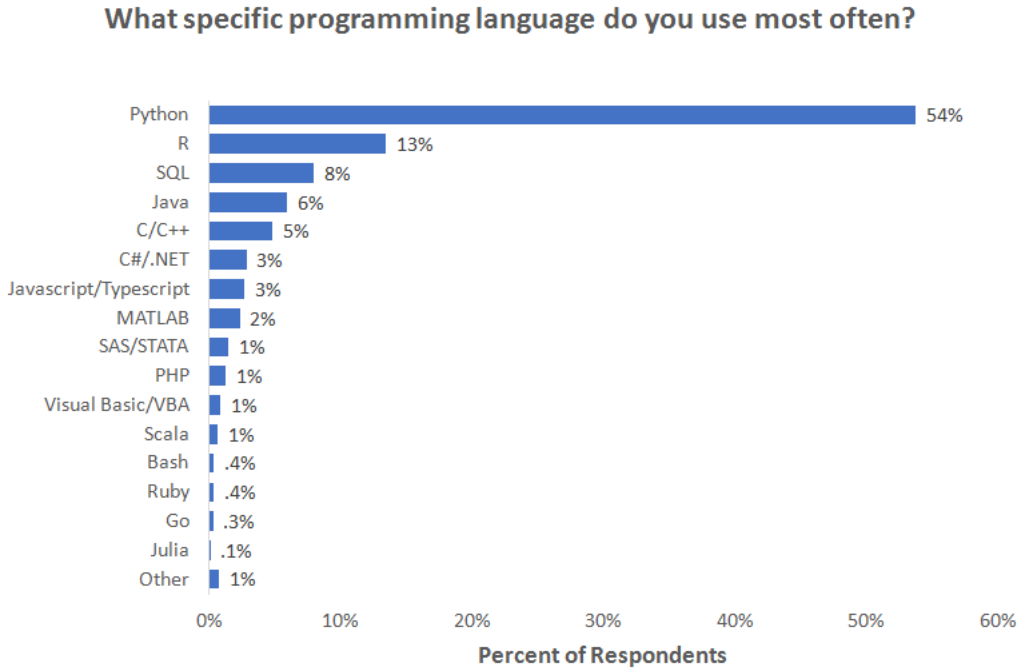
Other
- ElasticSearch
- R
- Julia
- Airflow

Most popular tools



Note: Data are from the 2019 Kaggle ML and Data Science Survey. You can learn more about the study here: https://www.kaggle.com/c/kaggle-survey-2019.
A total of 19717 respondents completed the survey; the percentages in the graph are based on a total of 14762 respondents who provided an answer to this question.

BUSINESS BROADWAY
DATA SCIENCE | CUSTOMER ANALYTICS | MACHINE LEARNING

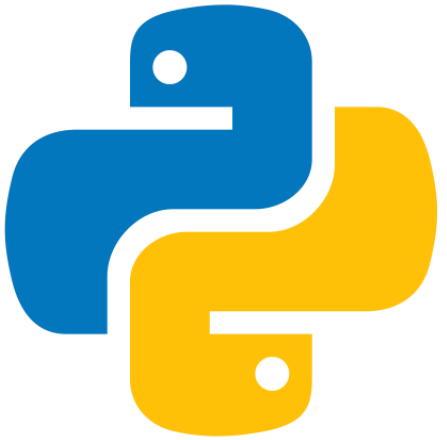


Note: Data are from the 2018 Kaggle ML and Data Science Survey. You can learn more about the study here: http://www.kaggle.com/kaggle/kaggle-survey-2018.
A total of 23859 respondents completed the survey; the percentages in the graph are based on a total of 15222 respondents who provided an answer to this question.

BUSINESS BROADWAY
DATA SCIENCE | CUSTOMER ANALYTICS | MACHINE LEARNING

Python

"*Python is powerful... and fast; plays well with others; runs everywhere; is friendly & easy to learn; is Open.*" **python.org**

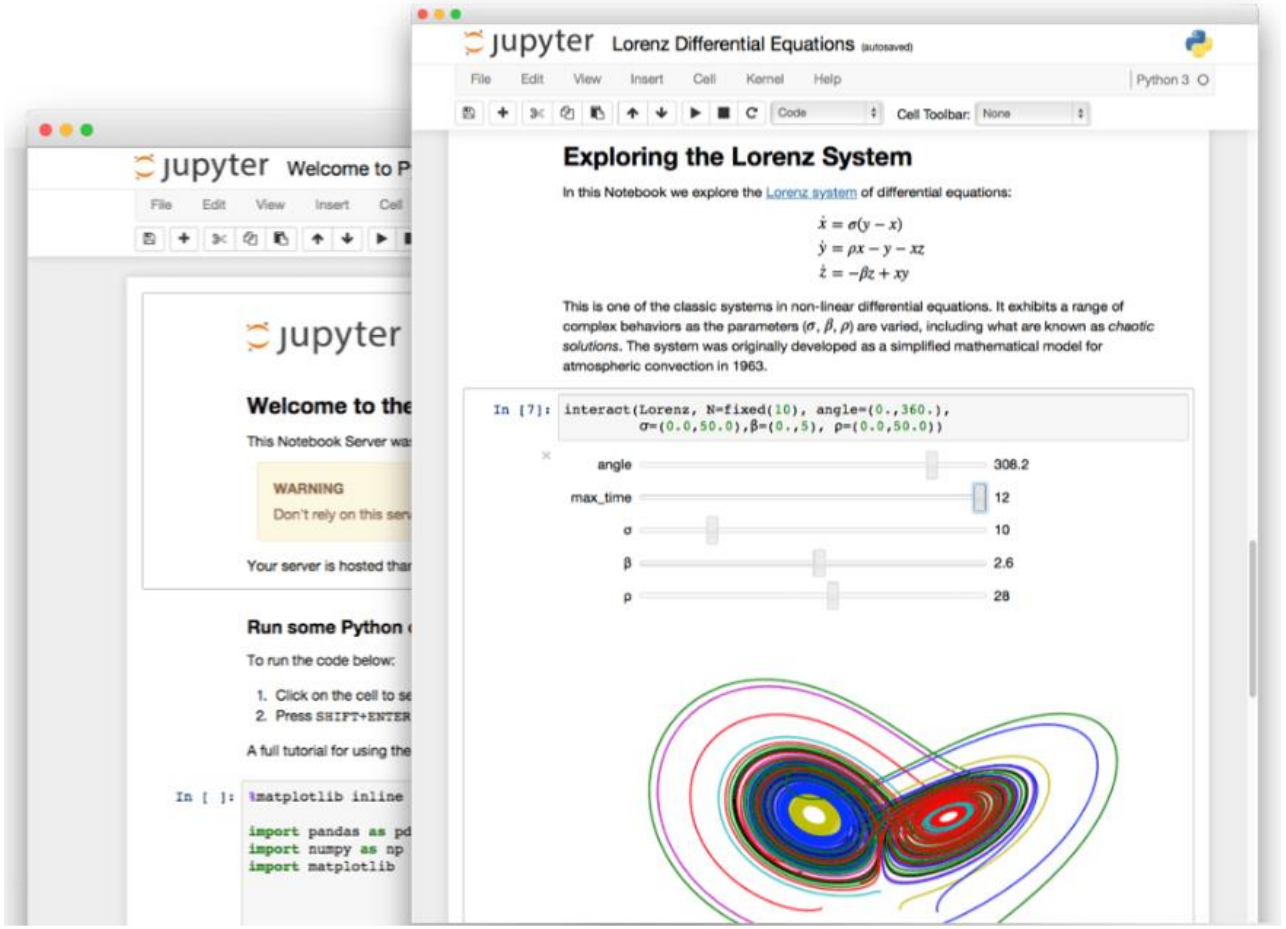


- High level programming language with strong support for data science. Created in 1991. Philosophy of code readability. Open source.
- Dynamic typing, automatic memory management. Procedural, object oriented, functional.
- IDEs (Spyder, PyCharm, VSCode, ...)
- Large community

IDEs - Jupyter notebooks / Jupyter lab

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text.

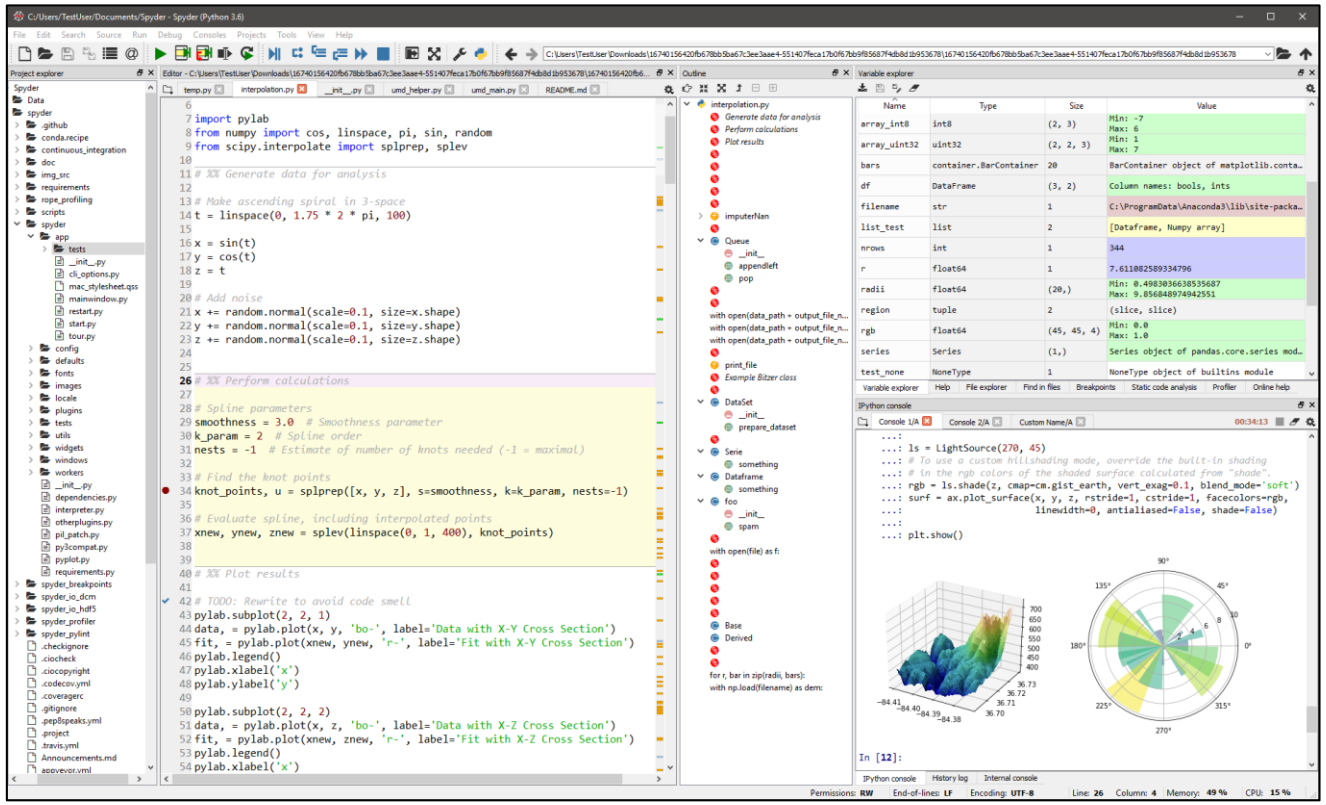
https://jupyter.org/



https://jupyter.org/

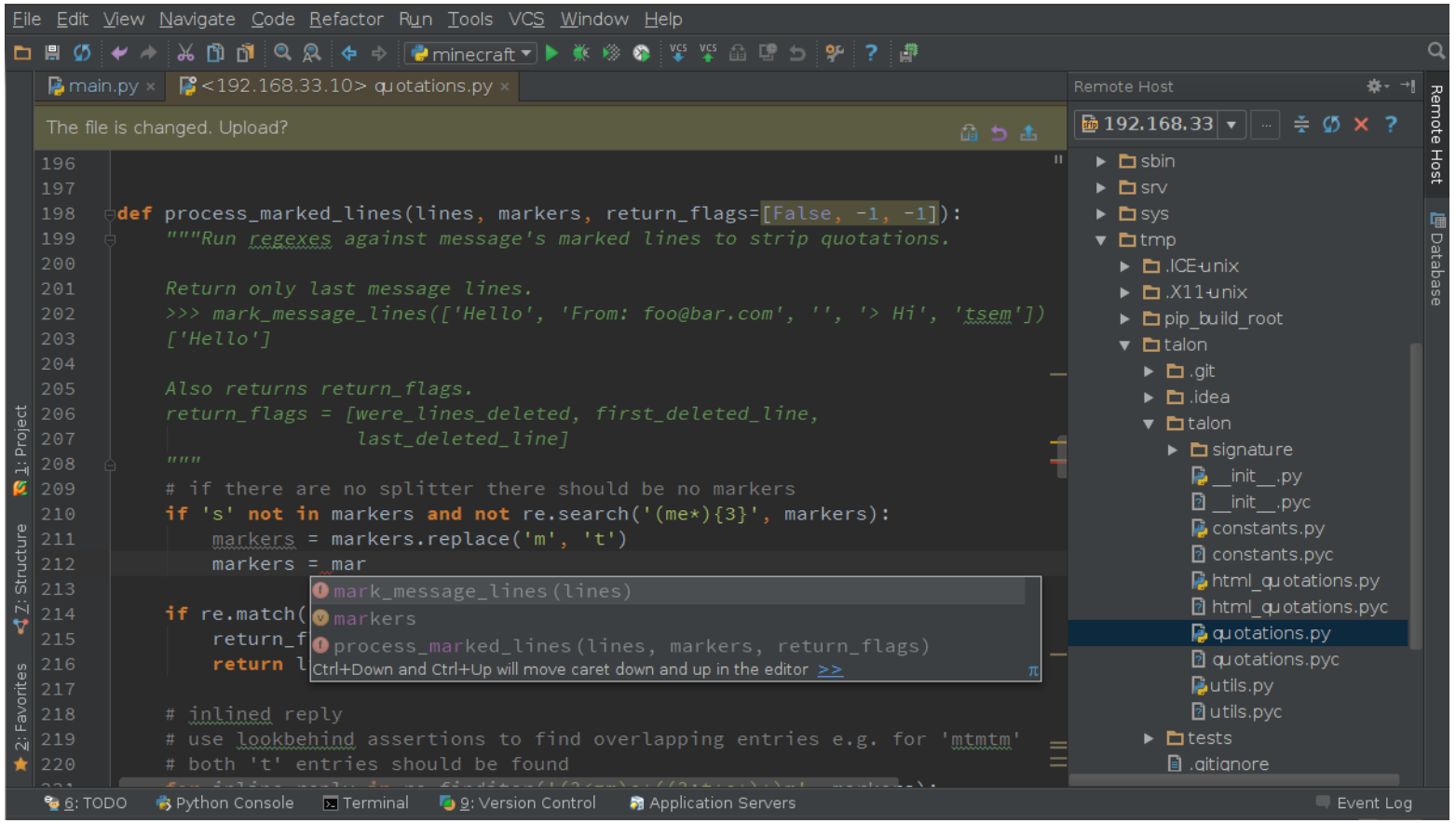
IDEs – Spyder

- Scientific Python IDE
- Similar to RStudio
- Loved by R & Python bilingual data scientists
- Part of Anaconda distribution
- Version 4 released in 2020 brining in many new features



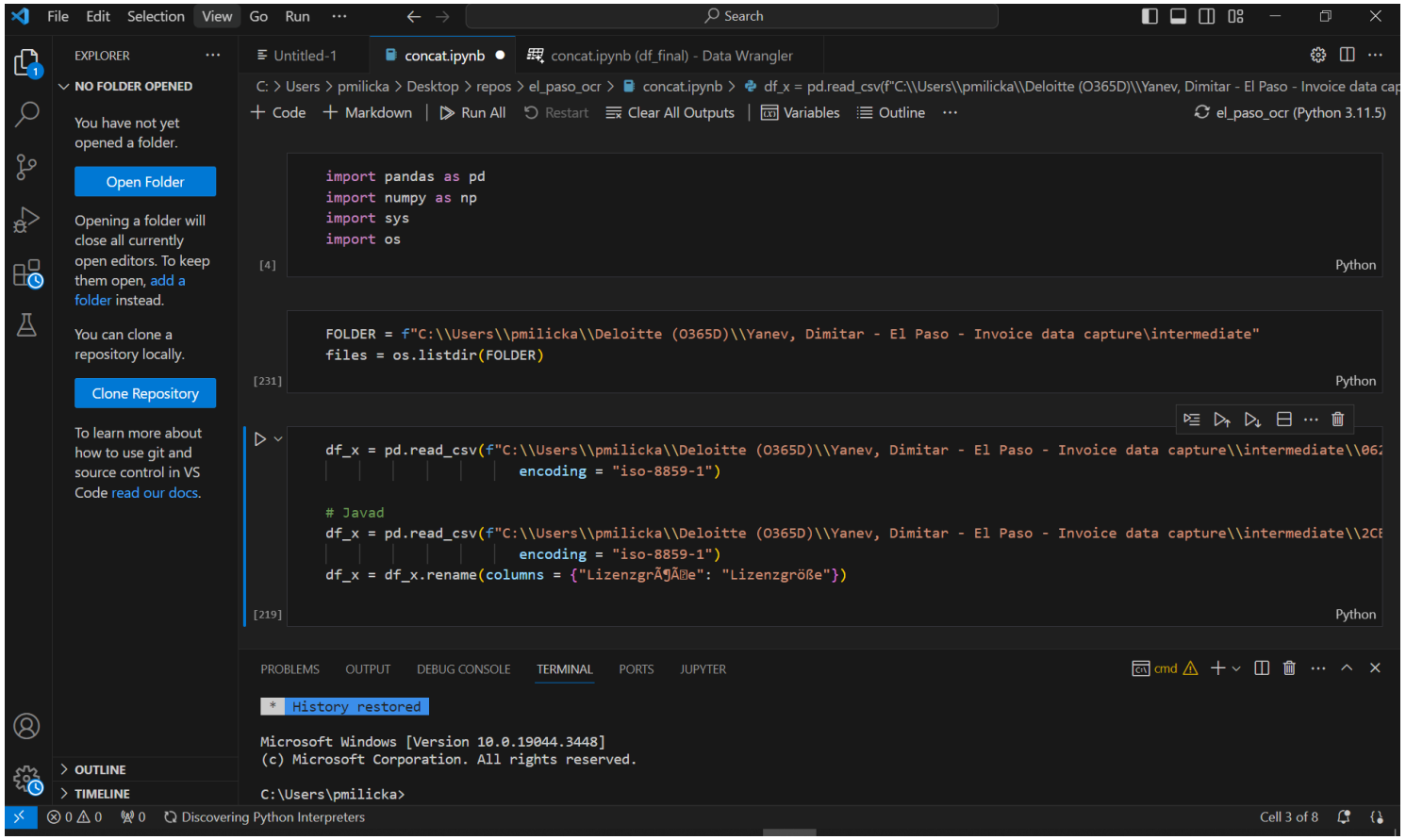
IDEs – PyCharm

- Rich IDE made by JetBrains (Czech company)
- Free and Professional edition
- Mainly focused for software developers (but it has a scientific mode similar to Spyder)



IDEs – VSCode

- Open Source
- Versatile, options are limitless as you can download extensions for remote development, notebook display, etc.
- Git integration
- Add-ons – cloud, Data Wrangler, Jupyter notebooks, Code Completion

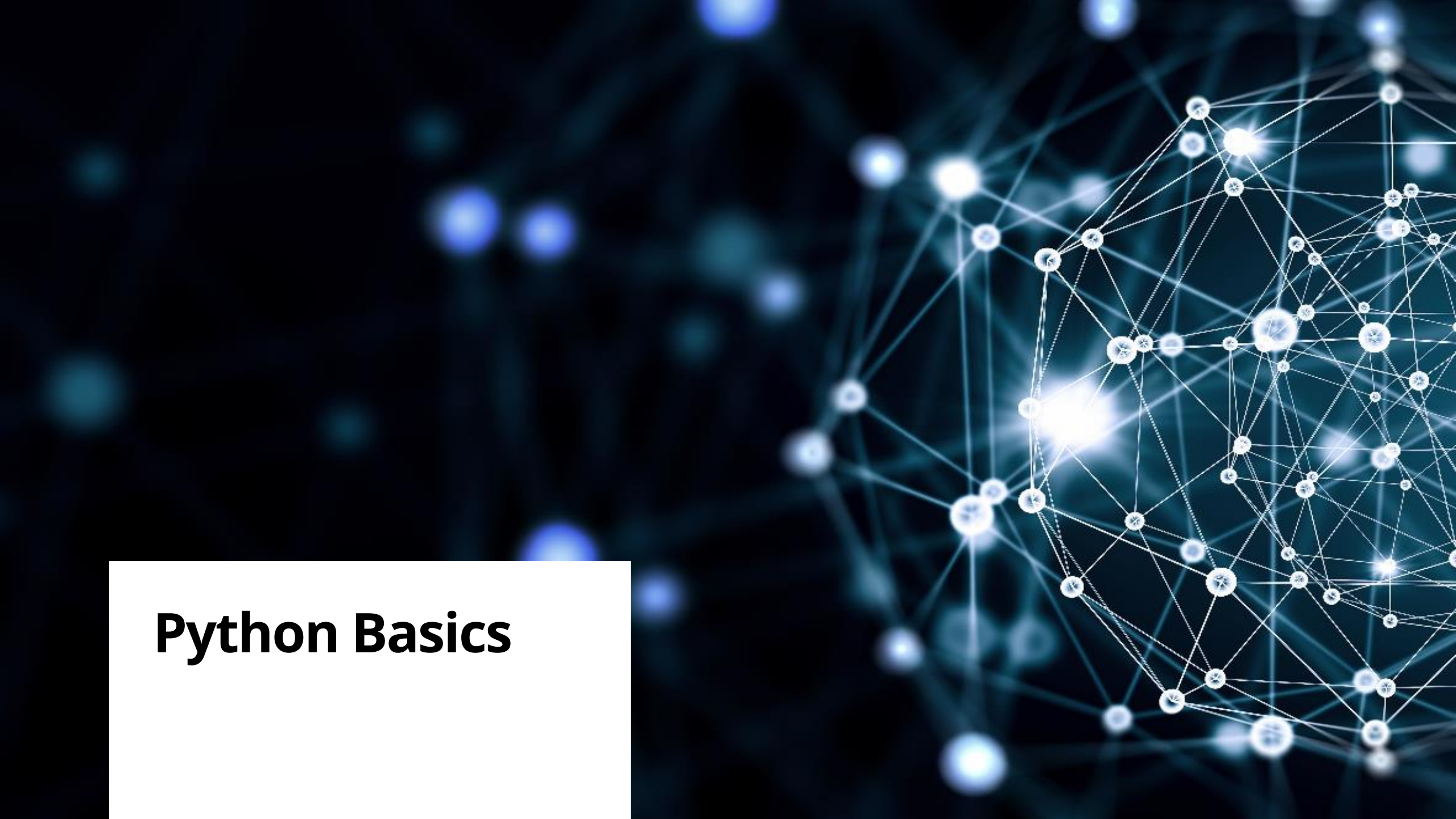


Python core Data Science packages

- Python can be augmented by using packages
- On 2020-30-07, the PyPI package repository features 253,532 packages
- Packages for everything – Web development (Django, Flask, ...), Natural Language Processing (gensim, spacy, nltk, ...), Web scraping (scrapy, selenium, BeautifulSoup...) etc.

Anaconda and Python environments

- Installing your own python environment
  - https://www.anaconda.com/
- Conda environments
  - `conda env list`
  - `conda create -n=<environment_name> python=3.8`
  - `conda activate <environment_name>`
- Package management
  - `pip install <package_name>`
  - `pip freeze`
  - `(conda install <package_name>)`



Python Basics

Python Basics hands-on

1. Open the D1_Python_101.ipynb notebook.

Version control systems – Git

Which file is the current one?

Version control systems – Introduction

A version control system, or VCS, tracks the history of changes as people and teams collaborate on projects together. As the project evolves, teams can run tests, fix bugs, and contribute new code with the confidence that any version can be recovered at any time. Developers can review project history to find out:

- Which changes were made?
- Who made the changes?
- When were the changes made?
- Why were changes needed?

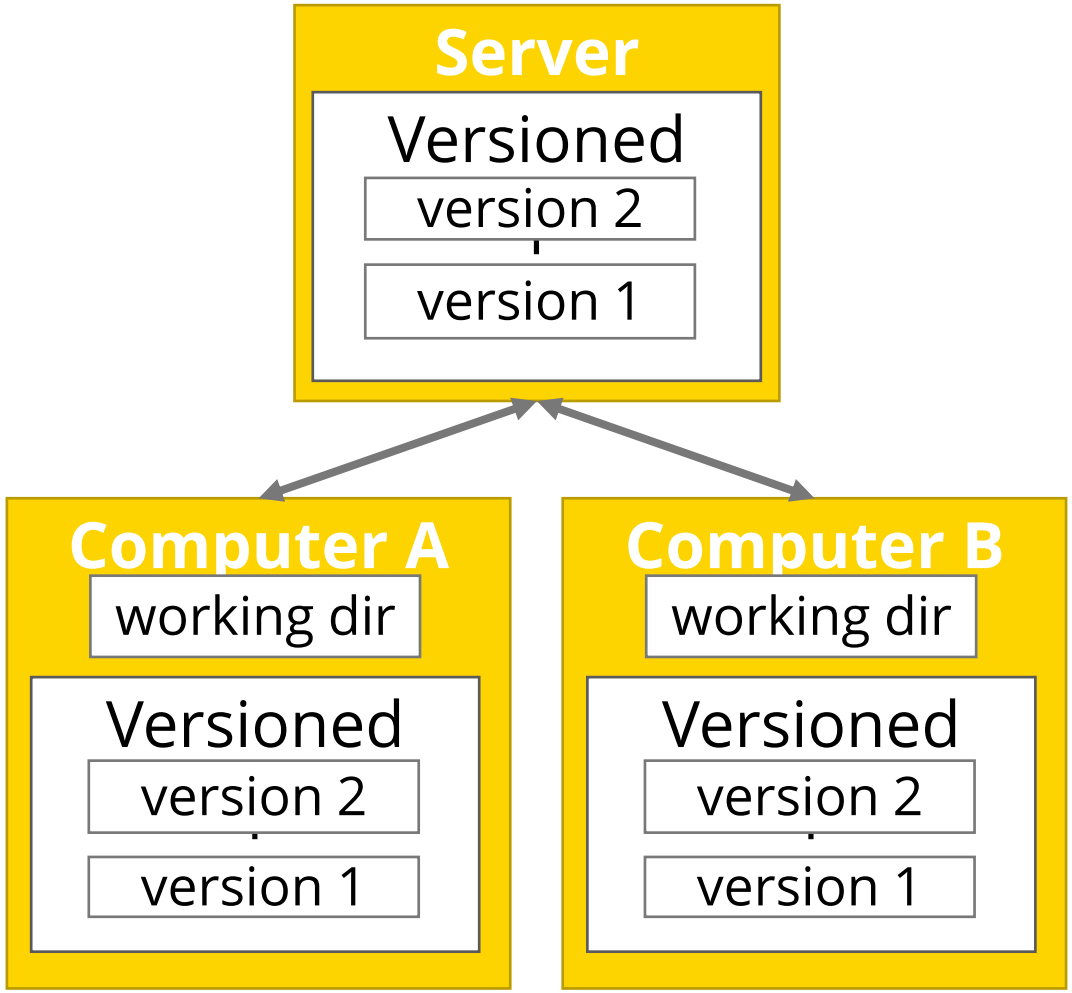
Version control systems – Benefits

- Protects source code from both catastrophe and the casual degradation of human error and unintended consequences.
- Allows work parallelization
- Helps organize changes and repairs in the code and document why the changes were made
- No need for file suffixing, all changes are backed by commit messages stating what was done and why.

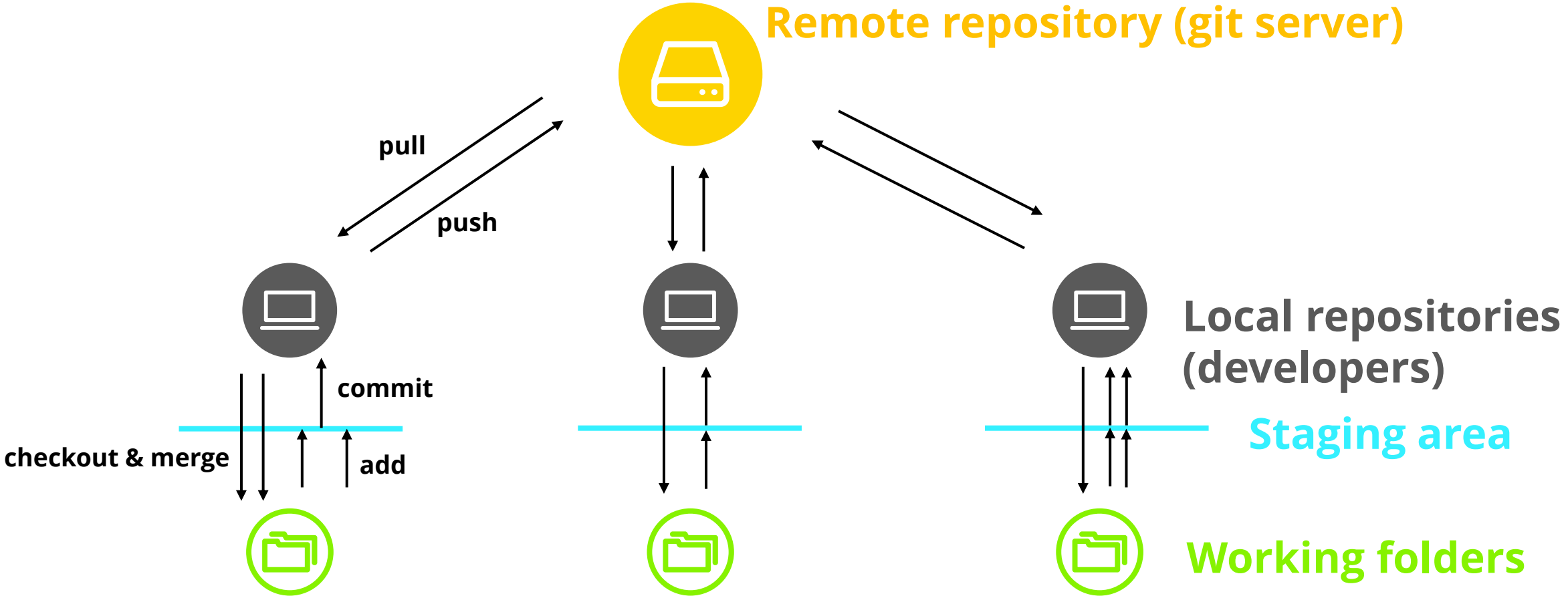
Most common version control systems are **git** and **SVN**.

Git – Introduction

- Created by author of Linux
- Repository on a server – interact with the remote by **clone**, **pull** and **push**
- **Local repository** is a complete copy of the remote server
- Many of the operations are done locally:
  - change branches
  - commit changes to the local repostiory
  - versioning
- Pushing the local repostiory makes the versioning visible for the central repostiory

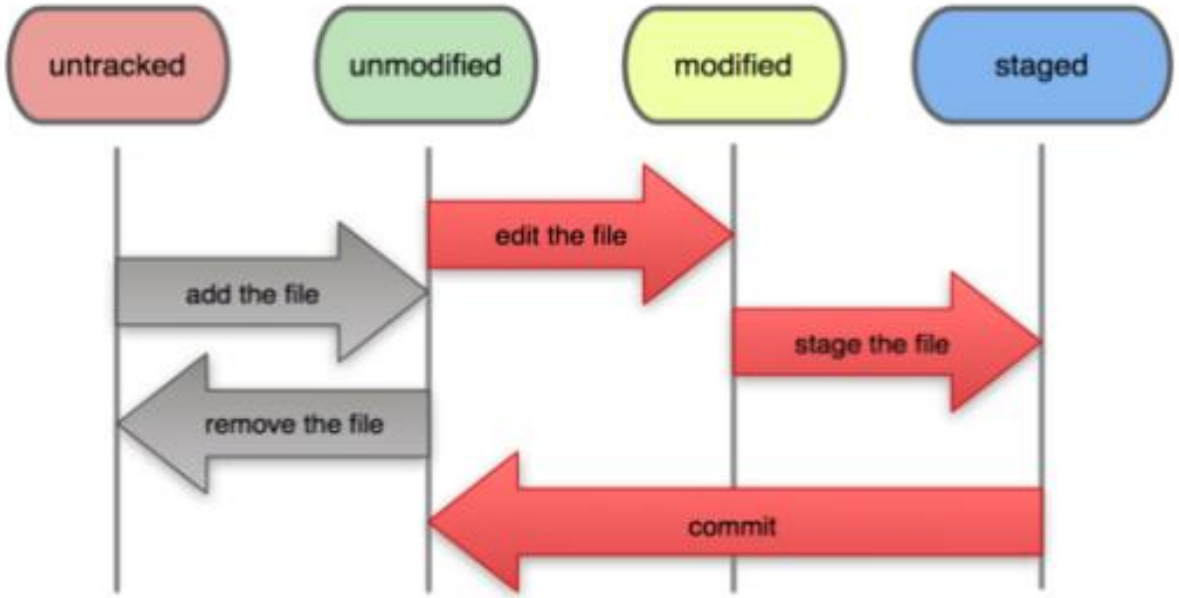


Git Schema



Local operations

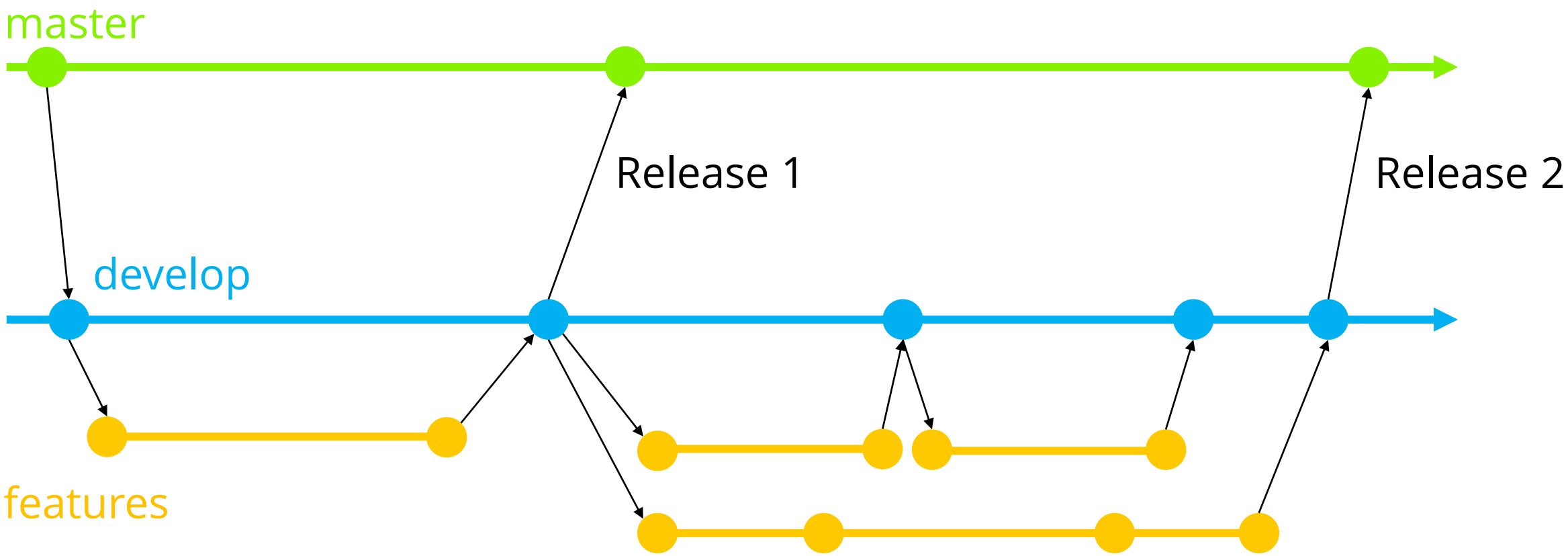
- **Add files** (git add) to tell git to version them
- **Stage files** create a snapshots of the files in he staging area
- **Commit** takes the files in the staging area and stores the snapshot permanently to the git repostiory



Branching

- Commits are associated with a branch.
- Default git branch is master.
- Branches are created from different branch (basically a copy)
- Branches allow to cluster commits belonging together (e.g. developers trying to implement specific feature vs others fixing a bug)
- Branches can be merged together when the new feature is ready and quality is assured to mess up the production branch.
- When two developers independently make conflicting changes, is creates a merge conflicts showing the conflicting parts with possibilities of resolution (use mine/use theirs/...)

Branching



Branching – Merge conflict

- File that where the same part was modified in two branches will result in **merge conflict**.
- If git is unable to resolve the conflict it adds <<< >>> sections to indicate where the problem occured, so the developer can decide which code is still relevant and which is faulty.

```
<<<<<<< HEAD:index.html
<div id="footer">todo: message here</div>
=======
<div id="footer">
  thanks for visiting our site
</div>
>>>>>>> SpecialBranch:index.html
```

branch 1's version

branch 2's version

Remote operations

- **Clone** takes a repository from a server and makes a local repository at given destination

```
git clone git://git.kernel.org/pub/scm/.../linux.git my-linux
```

- **Push** takes the local changes and writes them to the server.

```
git pull origin master
```

- **Pull** gets most recent changes from the repository.

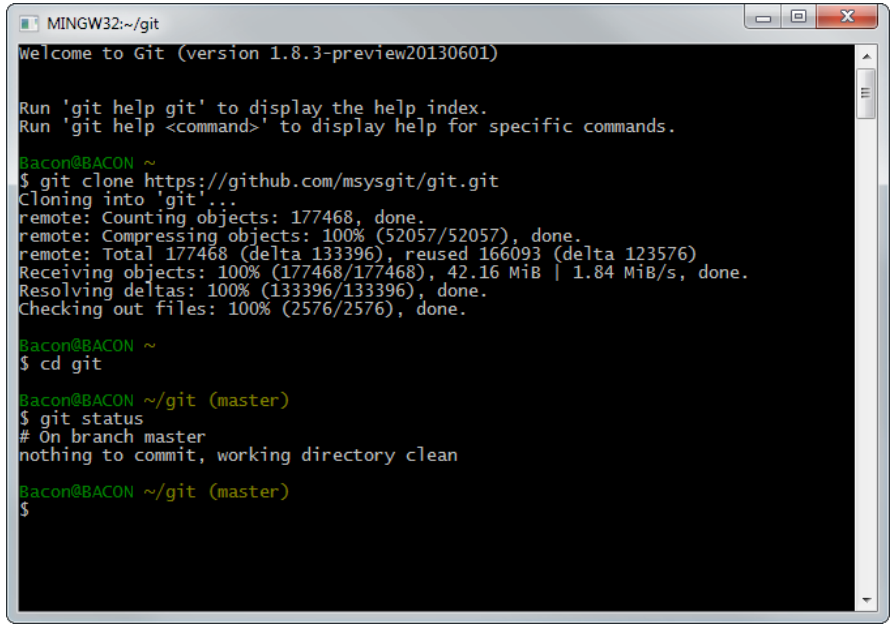
```
git pull origin master
```


Git – Workflow

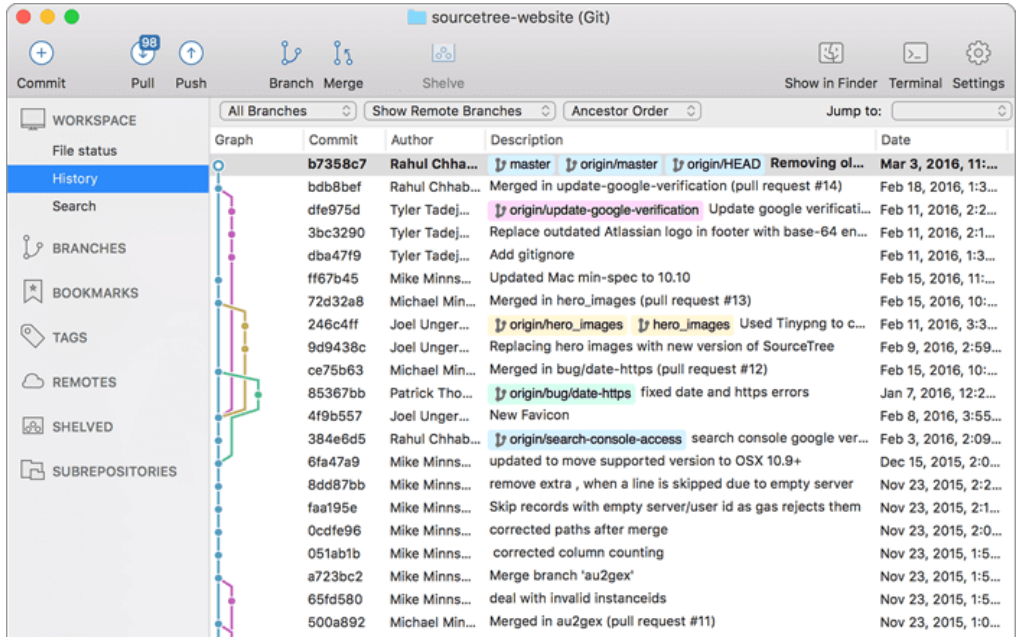
1. **Create a branch:** Topic branches created from the canonical deployment branch (usually master) allow teams to contribute to many parallel efforts. Short-lived topic branches, in particular, keep teams focused and results in quick ships.
2. **Add commits:** Snapshots of development efforts within a branch create safe, revertible points in the project's history.
3. **Open a pull request:** Pull requests publicize a project's ongoing efforts and set the tone for a transparent development process.
4. **Discuss and review code:** Teams participate in code reviews by commenting, testing, and reviewing open pull requests. Code review is at the core of an open and participatory culture.
5. **Merge:** Upon clicking merge, GitHub automatically performs the equivalent of a local 'git merge' operation. GitHub also keeps the entire branch development history on the merged pull request.

Git Bash vs Git GUI

- Less intuitive
- More technical approach
- Better control



- Easy to start
- No need to know commands
- Blackbox



Git with GUI demo

- Git server repository.
- Ticketing system.
- Tickets (Issues) can be connected with branches – the issues can be discussed in the tickets.
- Assigning tickets to people. Issues labeling.
- When the developing is complete a merge request is made and the code is assigned to a person non-active in the issue fix. The person reviews the code and merge the changes into the production branch.
- Milestones.
- CI/CD
- ...

Git with GUI demo

1. Create a gitlab repository
2. Clone the repository
3. Add files
4. Push
5. Issue with branch and merge request
6. Switch to branch, add a file and change some of the already present
7. Stage and commit
8. Switch between branches (how working directory behaves)
9. Push changes
10. Merge and close the issue/branch

Git – Essential Commands

GIT Command	Description
git init	Create empty Git repo in specified directory. Run with no arguments to initialize the current directory as a git repository
git clone <repo>	Clone repo located at <repo> onto local machine. Original repo can be located on the local filesystem or on a remote machine via HTTP or SSH.
git branch	List all of the branches in your repo. Add a <branch> argument to create a new branch with the name <branch>.
git checkout –b <branch>	Create and check out a new branch named <branch>. Drop the -b flag to checkout an existing branch.
git add <directory>	Stage all changes in <directory> for the next commit. Replace <directory> with a <file> to change a specific file.
git commit –m "<message>"	Commit the staged snapshot, but instead of launching a text editor, use <message> as the commit message.
git status	List which files are staged, unstaged, and untracked.
git pull <remote>	Fetch the specified remote's copy of current branch and immediately merge it into the local copy.
git push <remote> <branch>	Push the branch to <remote>, along with necessary commits and objects. Creates named branch in the remote repo if it doesn't exist.

GIT Cheat Sheet: https://www.atlassian.com/dam/jcr:8132028b-024f-4b6b-953e-e68fcce0c5fa/atlassian-git-cheatsheet.pdf

GIT Official Documentation: https://git-scm.com/docs/git-pull

Git in CMD demo

1. git status, git branch in the repository
2. modify a file, add a file
3. git status
4. git commit +msg
5. git push
6. show the results on

Tasks

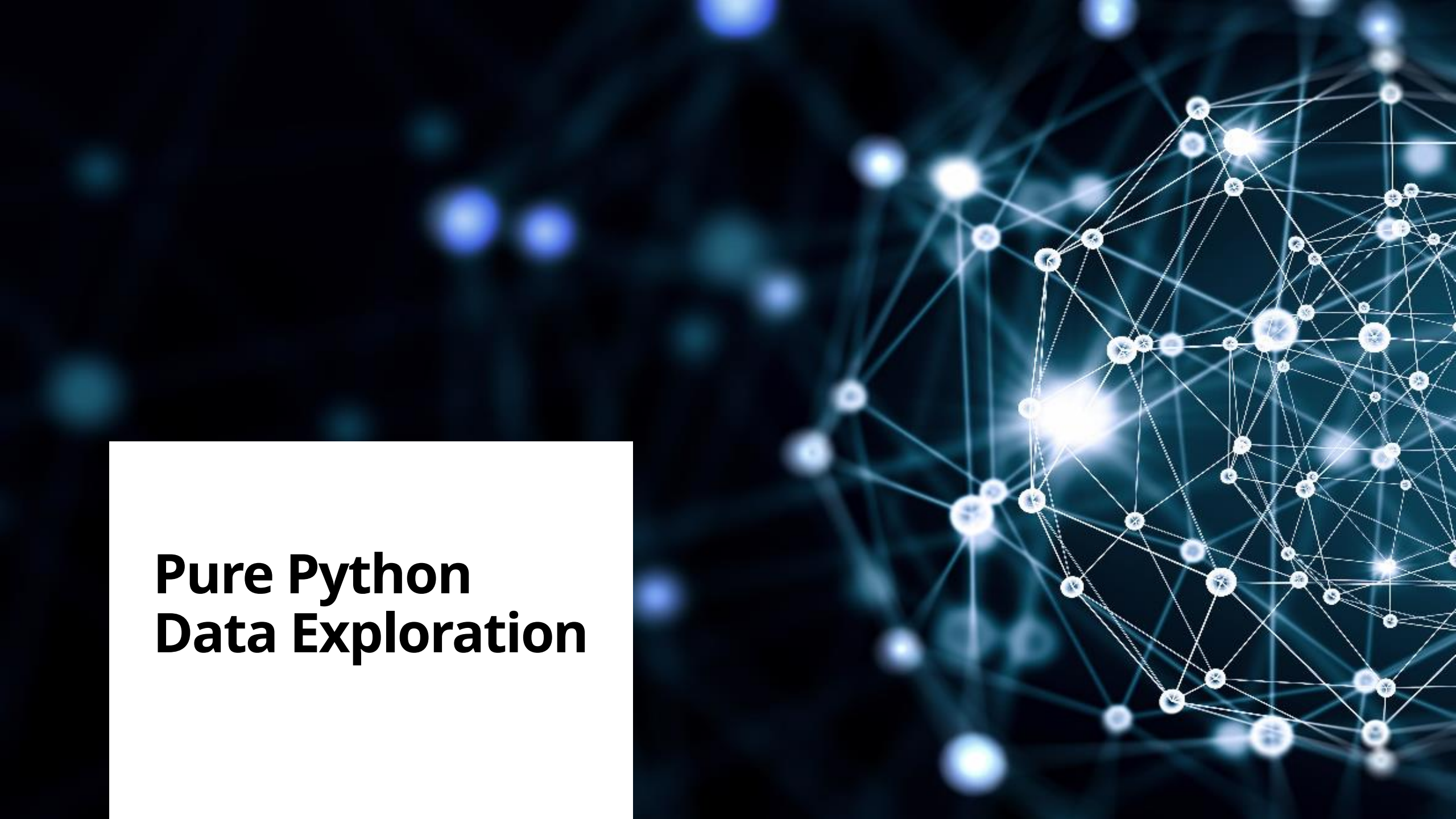
- Go to https://learngitbranching.js.org/ and complete the introduction sequence

Git – Takeaway message

- Git is good for you, even though it may seem that it only adds work, in the end having track of all the changes and being able to develop parallely outweights the drawbacks by far.
- Helps with Quality Assurance workflow.
- Git is a versioning system vs GitLab/GitHub are cloud repository storage with further functions.
  - You do not have to be on cloud to use git.
- You do not need to work in team to benefit from git – you can use it solely to version your work.
  - One branch, just commiting changes in time.
- It may seem overwhelming at first, but it is fairly simple once you get used to it. ☺

References:

- https://git-scm.com/
- https://courses.cs.washington.edu/courses/cse403/13au/lectures/git.ppt.pdf
- https://www.slideshare.net/naimlatifi/gitpresentation-140814102916phpapp01



Pure Python Data Exploration

Pure Python Data Exploration

1. Upload the google playstore datasets
2. Open jupyter notebook D1_google_apps_empty.ipynb

Python Libraries

Python Libraries introduction

1. Open the D1_Python_libraries.ipynb notebook

Home Assignment HA1

Pick one of the following

1. Scrape a website with product you want to of your choosing and output a list of relevant information about the offers (e.g. https://www.sbazar.cz/hledej/macbook?cena-dohodou=bez , https://www.mmreality.cz/nemovitosti/ , ...) (hint: BeautifulSoup)
2. For a list of stock symbols, get the last and future dividend payouts (e.g. https://www.nasdaq.com/market-activity/stocks/t/dividend-history) (hint: BeautifulSoup or find a free API)
3. Create a script (function) that states size of all files inside recursively and returns the path to the largest file. Raise error if given path is not a directory. Do this on local python installation in order to test it. (hint: os)
4. Any other ideas are welcome.

Save your script/notebok and send it to *pmilicka@deloitteCE.com* with subject **DSI_HA01_<surname>** for review.

Deloitte.