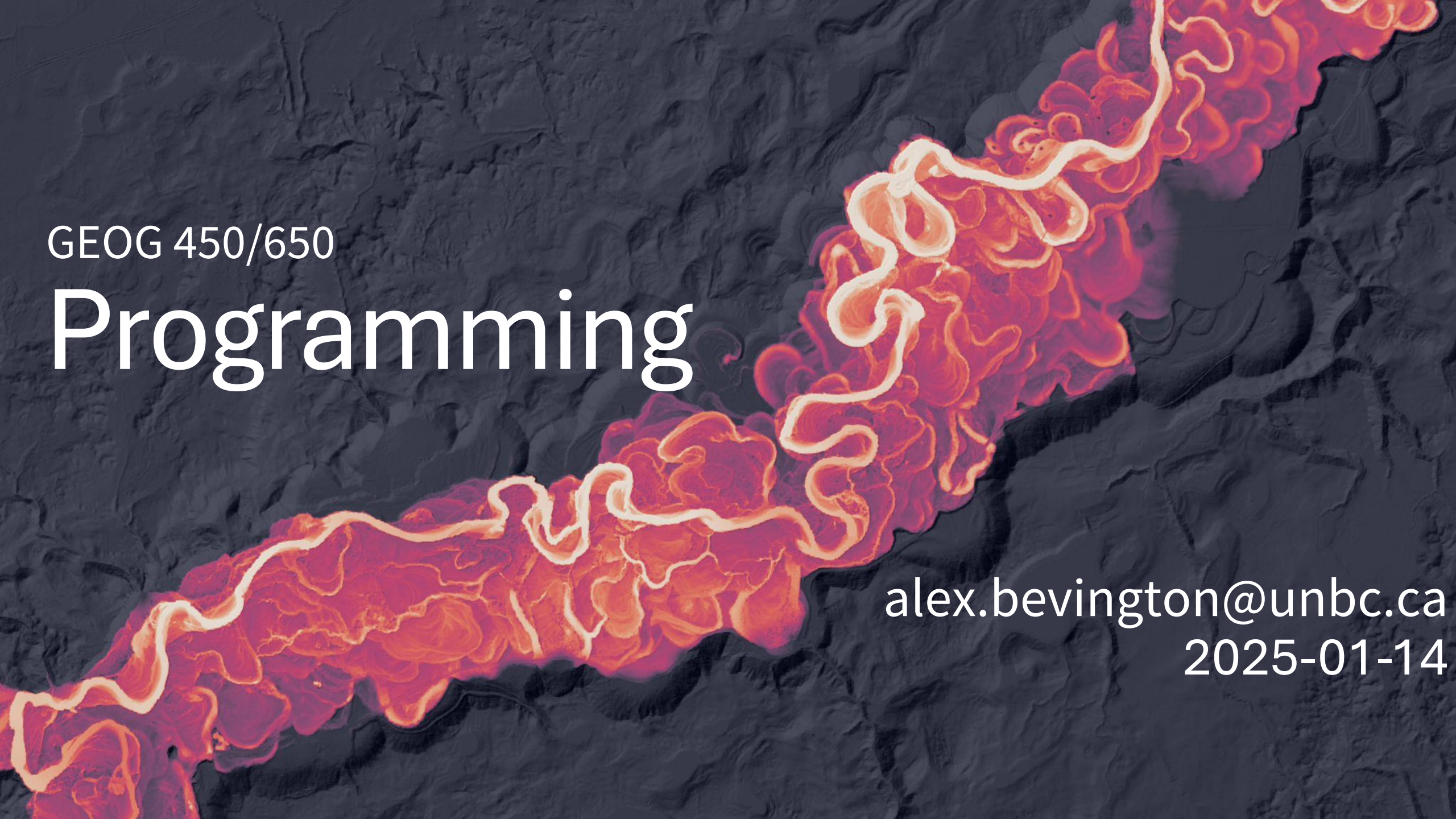


GEOG 450/650

Programming

alex.bevington@unbc.ca

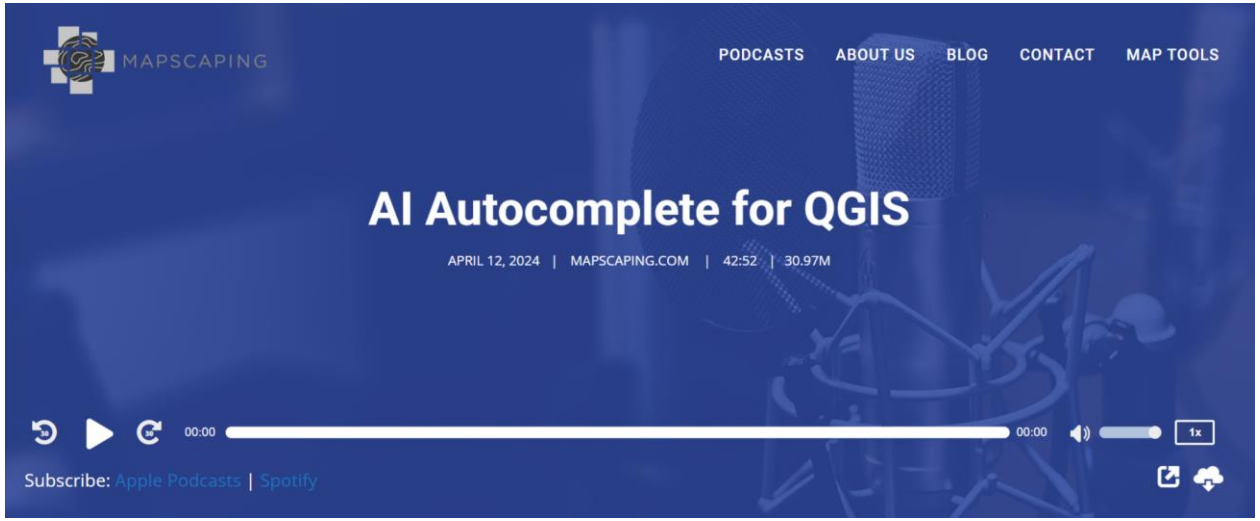
2025-01-14

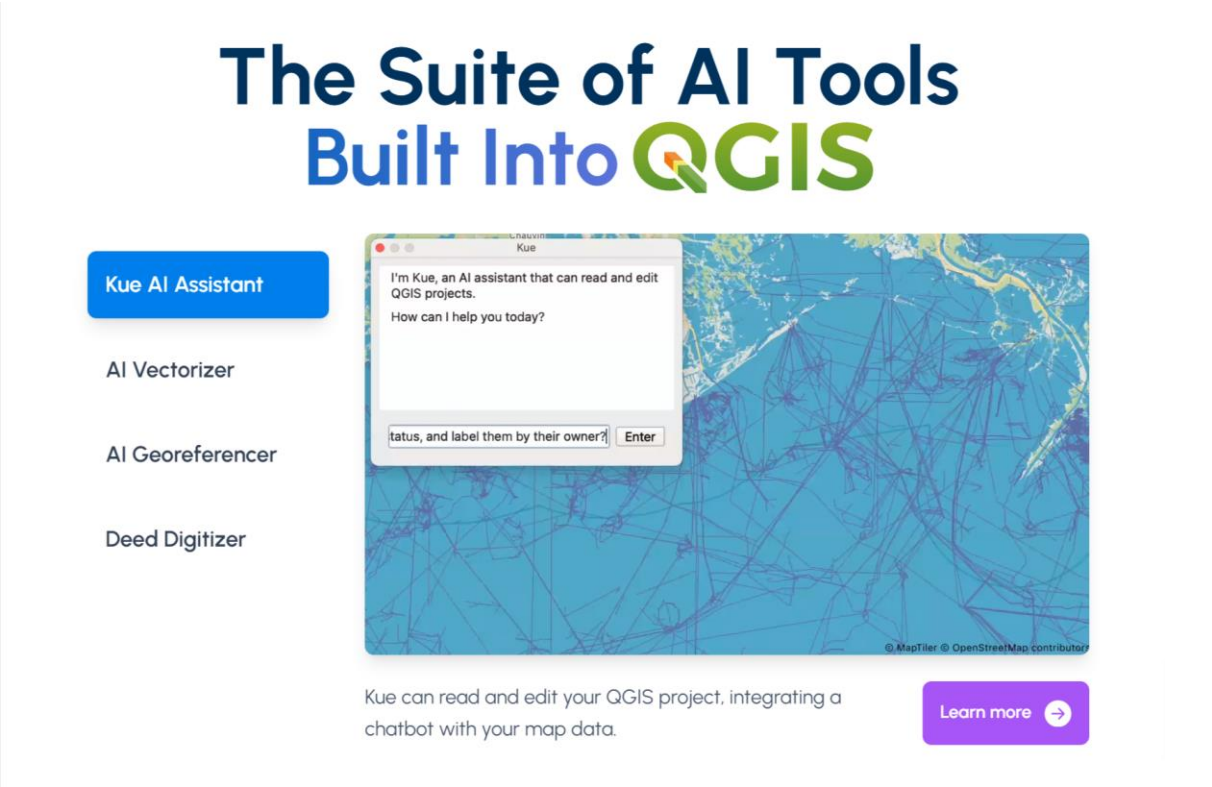
Housekeeping

- **Syllabus**: updates on course website
- **Reminders**:
  - Create (free) Google Earth Engine account by Thursday
  - Read Coetzee et al. (2020) by Thursday
- **Project** description will be handed out next Tuesday
- **Mid-term** will be based the guest speaker presentations

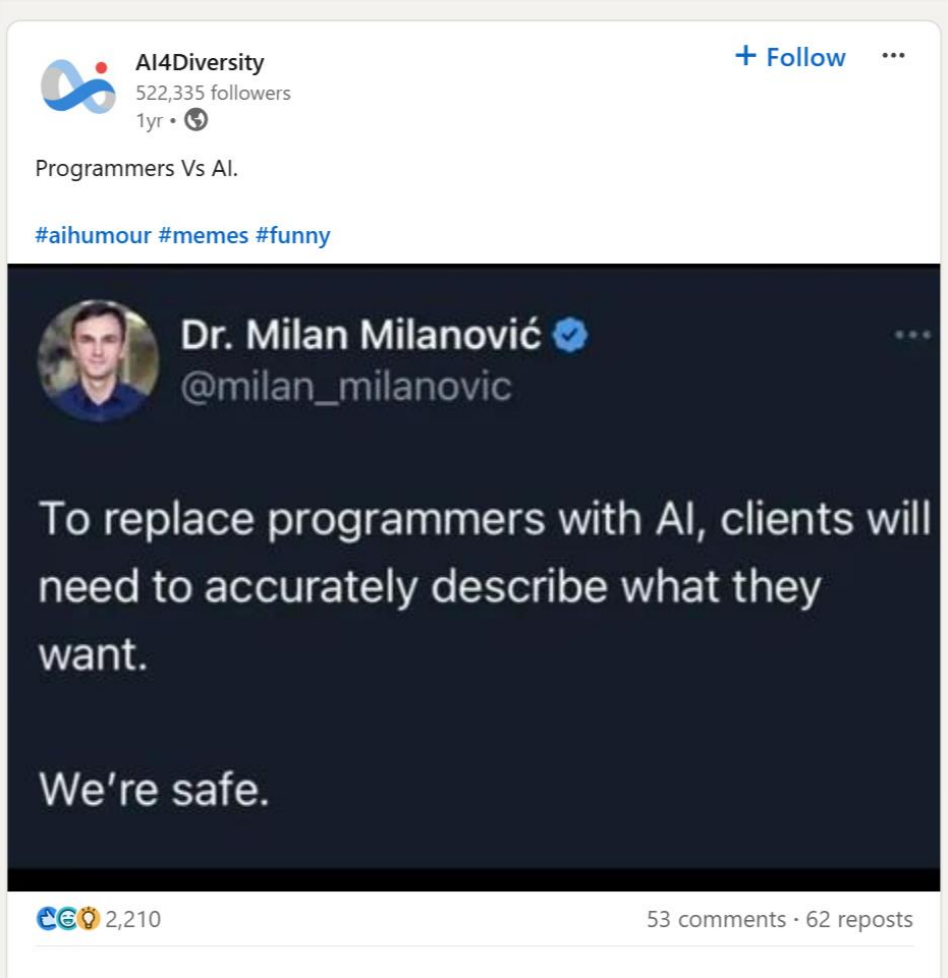
Podcast discussion

- Main points
- What's exciting
- Implications





Is it worth learning to code in 2025?



Reasons to learn coding

- Automate repetitive tasks
- Analyze large datasets
- Create reproducible workflows
- Widely applicable skill
- Many workflows don’t exist
- Cutting edge

Launcher | demo.ipynb

Code | Python (ox)

[1]:
```
import contextily as cx
import geopandas as gpd
```

Load a shapefile of california census tracts as a GeoDataFrame:

[2]:
```
gdf = gpd.read_file('./tl_2019_06_tract/')
gdf.shape  # output GeoDataFrame count of rows, columns
```

[2]: (8057, 13)

Filter the GeoDataFrame to retain only tracts in Southern California counties:

[3]:
```
socal = ['025', '029', '037', '059', '065', '071', '073', '079', '083', '111']
gdf = gdf[gdf['COUNTYFP'].isin(socal)]
gdf.shape
```

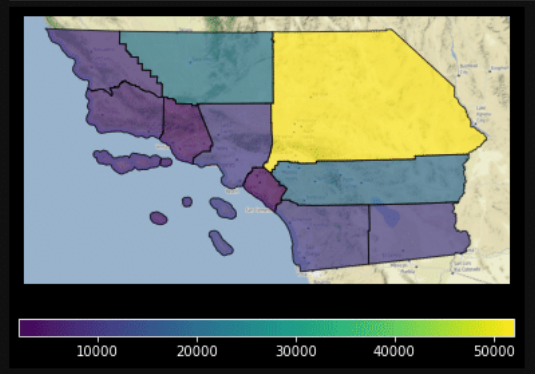
[3]: (4879, 13)

Dissolve the tracts into counties, summing numerical field, then project the counties to the CRS of the California State Plane Coordinate System zone 5:

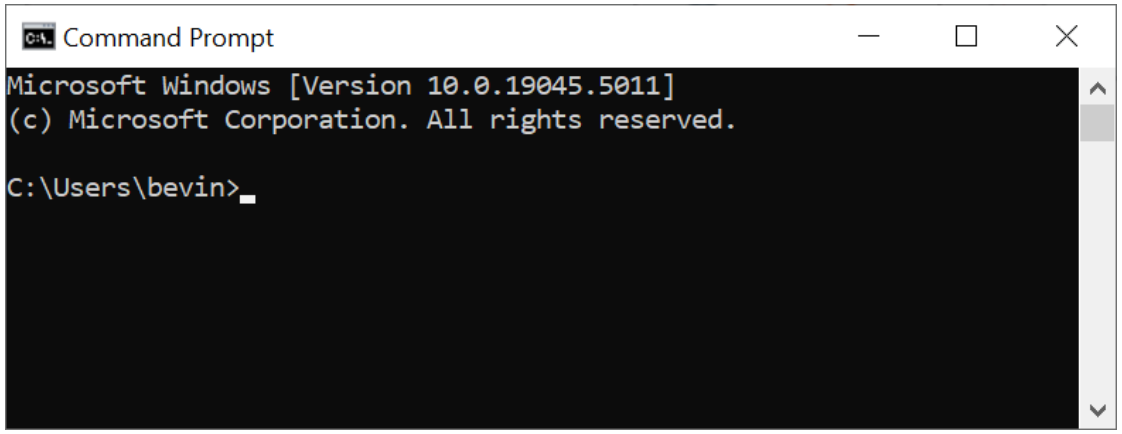
[4]:
```
counties = gdf.dissolve(by='COUNTYFP', aggfunc='sum')
counties = counties.to_crs('EPSG:3497')
```

Plot a choropleth map of the data, with counties colored by land area (km²):

[5]:
```
counties['ALAND'] = counties['ALAND'] / 1e6 # convert m^2 to km^2
ax = counties.plot(column='ALAND', cmap='viridis', edgecolor='k', alpha=0.7, legend=True)
cx.add_basemap(ax, crs=counties.crs.to_string()) # add basemap with contextily
_ = ax.axis('off')
```



Programming 101: Command line



- A text-based interface for interacting with your computer, allowing users to execute commands directly.
  - Prompt: Indicates the system is ready to receive commands.
  - Commands: Text instructions to perform actions.
  - Arguments: Additional options or data passed to commands.
- Advantages:
  - Fast and efficient for repetitive tasks.
  - Enables automation through scripts.
  - Direct access to system functions not always available via GUI.

Programming 101: Command line

- General Syntax:
  - command [options] [arguments]
- Examples:
  - List files and directories: **ls –l**
  - Change directory: **cd /path/to/directory**
  - Create a file: **touch filename.txt**
  - Useful for running software/scripts, etc.
- Tons of learning resources online (e.g. command line ninja!)

Overview of R

- Originally designed for statistics and data visualization
- Very popular with ecologists, hydrologists, biologists, etc.
- Less for geospatial development, but still very powerful.
- Key features:
  - Powerful data manipulation and statistical packages
  - Rich visualization libraries
- R Studio is the most popular IDE

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function | Addins | Project: (None)

Untitled1

Source on Save | Run | Source

```
1
```

1:1 (Top Level) R Script

Console Terminal Background Jobs

R 4.3.2 · C:/Users/bevin/Desktop/

```
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

>
```

Environment History Connections Tutorial

Import Dataset | 130 MiB | List

R | Global Environment

Data

.Last.value	List of 1

Files Plots Packages Help Viewer Presentation

Zoom | Export

Python

- General-purpose programming language
- Very popular in geospatial analysis
- Well integrated with QGIS, ArcGIS, Earth Engine, etc.
- Key features:
  - Versatility (data science, web development, etc.)
  - Extensive libraries and frameworks
- Many IDEs (Jupyter, VS Code, PyCharm, Spyder, etc.)

Untitled17 - Jupyter Notebook

localhost:8888/notebooks/Untitled17.ipynb?kernel_name=python3

jupyter Untitled17 Last Checkpoint: a few seconds ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted Python 3

Run Code Voilà

In [ ]:

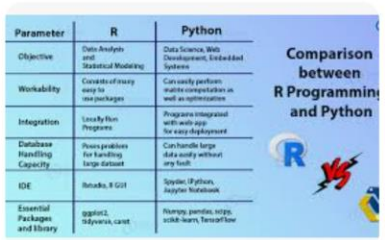
Other languages, for example…

- **General-Purpose**: Java, C, C++, C#, Rust
- **Web**: JavaScript, TypeScript, PHP, Ruby, HTML & CSS
- **Data Science**: Julia, MATLAB, SAS
- **Database and Query Languages**: SQL, MQL, etc.

Google r vs python

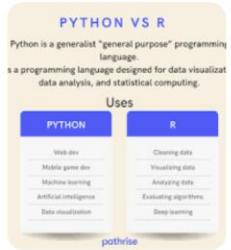
All Images Videos News Shopping Web Books More Tools Saved

Julia | Difference between | Python programming language | Trends | Data visualization | R programming | Salary | Artificial intelligence | Statistical analysis | Deep lear

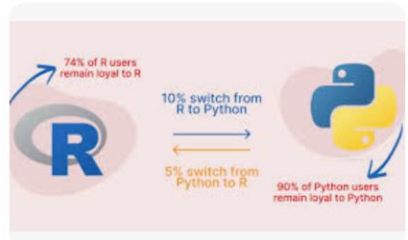


DataFlair
R Vs Python - The most trending debat…

GeeksforGeeks
R vs Python - GeeksforGeeks



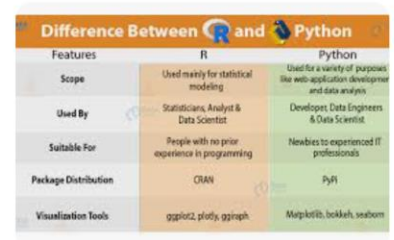
Pathrise
Python vs R: Data sc…



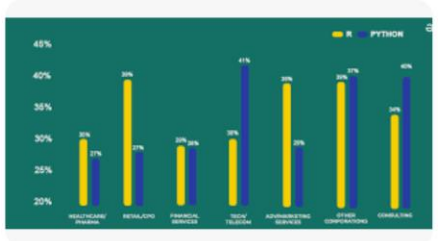
Extern Labs
R vs Python: Which is Better For Data ...

phoenixNAP
R vs. Python - A D…

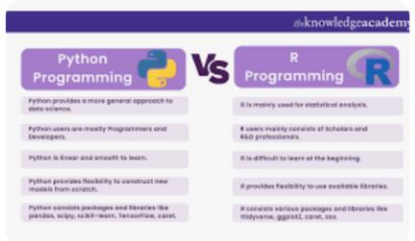
Imaginary Cloud
R vs Python: The Data Science langu…



DataFlair
R vs Python for Data Science - And th…

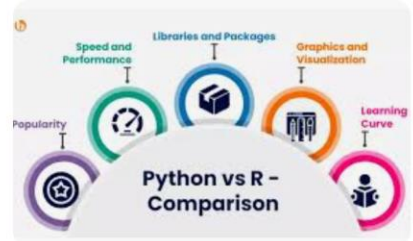


GeeksforGeeks
R vs Python - GeeksforGeeks



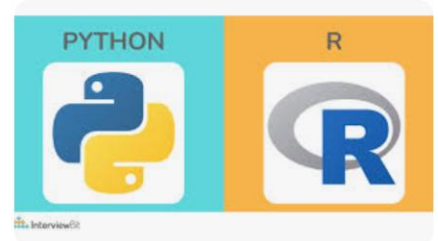
The Knowledge Academy
R Programming Language vs. Python ...

StrataScratch
Python vs R for Data Science ...

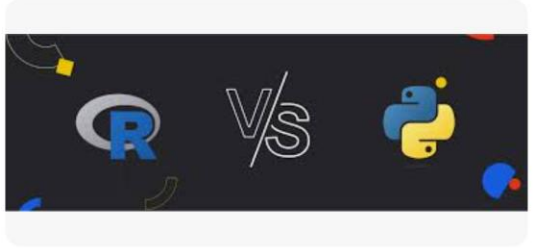


Bacancy Technology
Python vs R - The Right Choice For Data ...

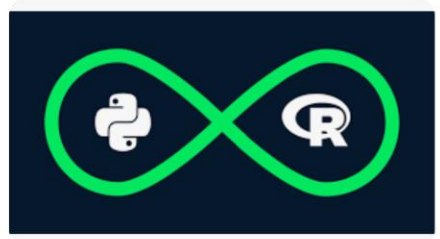
Towards Data Science
Python vs R: The Basics. An as…



InterviewBit
Python Vs R: Know The Difference ...



Extern Labs

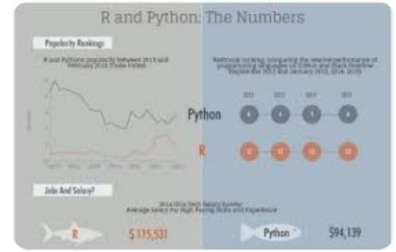


DataCamp

Medium

AlmaBetter

Business Science



KDnuggets

We’re not so different!

- **R**

```
# Hello, R!
print("Hello, world!")

install.packages(“dplyr”)
library(dplyr)
```

- **Python**

```
# Hello, Python!
print("Hello, world!")

pip install numpy
import numpy
```

Functionality	Python Package(s)	R Package(s)
Data Manipulation	*pandas*	*dplyr, data.table*
Data Visualization	*matplotlib, seaborn, plotly*	*ggplot2, plotly, lattice*
Statistical Analysis	*statsmodels, scipy.stats*	*stats, lm, MASS*
Machine Learning	*scikit-learn, tensorflow, keras*	*caret, mlr3, xgboost, h2o*
Deep Learning	*tensorflow, keras, PyTorch*	*tensorflow, keras, torch*
Geospatial Analysis	*geopandas, shapely, rasterio*	*sf, terra, stars*
Google Earth Engine	*geemap*	*rgee*
Time Series Analysis	*statsmodels.tsa, prophet*	*forecast, zoo, xts*
Database Interaction	*SQLAlchemy, sqlite3, psycopg2*	*DBI, RSQLite, Rpostgres*
Parallel Computing	*dask*	*parallel, future*
Reproducible Reports	*jupyter*	*knitr, rmarkdown*
Interactive Dashboards	*streamlit*	*shiny*

Category	R Data Types	Python Data Types
Numeric	*Integer, numeric*	*int, float*
Character/String	*character*	*str*
Logical/Boolean	*logical (TRUE, FALSE)*	*bool (True, False)*
Vectors	*vector (e.g., c(1, 2, 3))*	*numpy.array*
Lists	*List*	*list*
Matrices	*Matrix*	*numpy.array*
Data Frames	*data.frame, tibble*	*pandas.DataFrame*
Factors/Categorical	*factor*	*pandas.Categorical*
Null/Empty	*NULL, NA, NaN*	*None, NaN, pd.NA*
Dates and Times	*Date, POSIXct, POSIXlt*	*datetime.date, datetime.datetime*

Functions: Example add two numbers

R

```r
add_numbers <- function(a, b){
    return(a + b)}
add_numbers(5, 3)
> 8
```

Python

```python
def add_numbers(a, b):
    return a + b
add_numbers(5, 3)
> 8
```

Logical operators

Operation	R	Python
Equal to	==	==
Not equal to	!=	!=
Greater than	>	>
Less than	<	<
Greater or equal	>=	>=
Less or equal	<=	<=
Logical AND	&	and
Logical OR	`	`
Logical NOT	!	not

R

```
x <- 10
y <- 20

if (x > 5 & y < 30) {
        print("Both are true")
} else {
        print("At least one is false") }
```

Python

```
x = 10
y = 20

if x > 5 and y < 30:
        print("Both are true")
else:
        print("At least one is false")}
```

Loops

- **R**

```
my_list <- c(1, 2, 3, 4)

for (item in my_list) {
    print(item)}
```

- **Python**

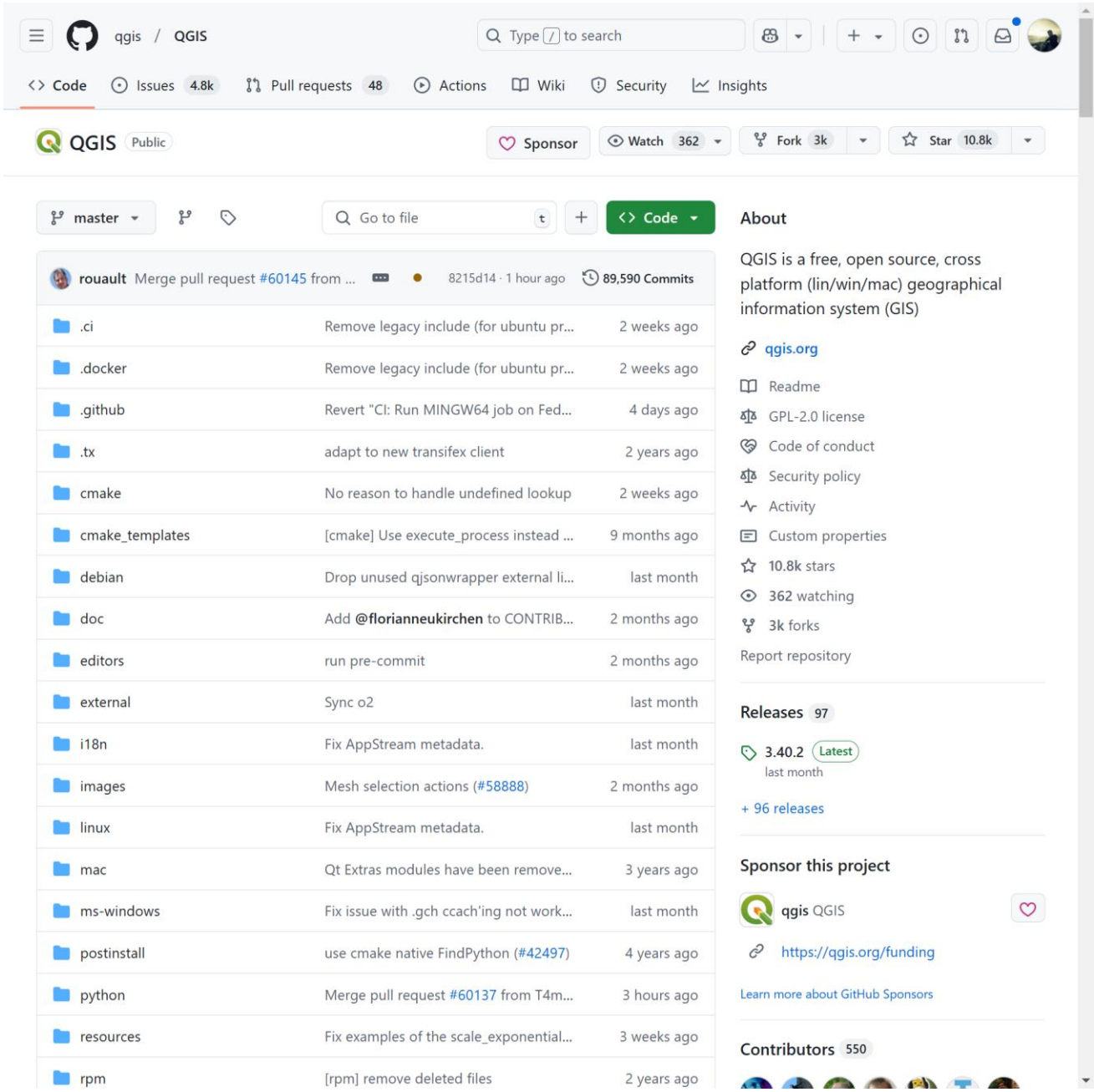
```
my_list = [1, 2, 3, 4]

for item in my_list:
    print(item)
```

Iterating over elements

- In both R and Python, **lapply** and **map** are tools for functional programming that allow you to apply a function to each element in a list or vector.
- They are often used as an alternative to traditional loops because they can be more concise and are often faster.

Version control with `git`

- Tracks changes in code
- Merge code from many users
- Parallel development without impacting the main repo
- Logs changes over time (good for debugging and review)



Reproducible workflows demo

- Example from BC ungauged basins project:
  - https://dankovacek.github.io/bcub_demo/notebooks/1_Getting_Started.html
- geemap package in python
  - https://geemap.org/notebooks/18_create_landsat_timelapse/#create-landsat-timeseries