**1**

# Importing pandas

**import** pandas **as** pd

# Loading in the data

pulls\_one = pd.read\_csv('datasets/pulls\_2011-2013.csv')

pulls\_two = pd.read\_csv('datasets/pulls\_2014-2018.csv')

pull\_files = pd.read\_csv('datasets/pull\_files.csv')

Wat het script hierboven doet is pandas importeren en de csv bestanden opvragen en laten zien.

Dit wist ik nog van de datacampcursus.

2

# Append pulls\_one to pulls\_two

pulls = pulls\_two.append(pulls\_one, ignore\_index=True)

Voegt 2 tabellen samen.

# Convert the date for the pulls object

pulls['date'] = pd.to\_datetime(pulls['date'], utc=True)

Hiermee worden de tijdstippen in de kolommen om worden gezet naar dezelfde tijdzones.

3

data = pulls.merge(pull\_files, on='pid')

2 tabellen worden samengevoegd en onder een nieuwe naam opgeslagen.

Brond: https://www.w3schools.com/python/pandas/default.asp

4

%matplotlib inline

# Create a column that will store the month

data['month'] = data['date'].dt.month

# Create a column that will store the year

data['year'] = data['date'].dt.year

# Group by month\_year and count the pull requests

counts = data.groupby(['year', 'month'])['pid'].count()

# Plot the results

counts.plot(kind='bar', figsize = (12,4))

Deze begreep ik niet helemaal dus heb hierbij de hints van Datacamp gebruikt. Wat het script doet is kolommen creëren die de waardes van opgevraagde rijen omzet in de nieuwe kolom en hernoemt.

De groupby zorgt ervoor dat deze samen worden gevoegd en zo worden weergeven. De plot zorgt ervoor dat er en diagram gemaakt wordt.

https://app.datacamp.com

5

# Required for matplotlib

%matplotlib inline

# Group by the submitter

by\_user = data.groupby('user').agg({'pid': 'count'})

# Plot the histogram

by\_user.hist()

Dit script creert een histogram van het aantal pull aanvragen dat is gedaan door elke user in de dataset.

https://app.datacamp.com

6

# Identify the last 10 pull requests

last\_10 = pulls.sort\_values(by = 'date').tail(10)

last\_10

Deze selecteert de 10 laatste pull requests en slaat ze op onder de variabele last 10 en worden op datum geordend.

# Join the two data sets

joined\_pr = pull\_files.merge(last\_10, on='pid')

Het combineert de 'pull\_files'-dataset met de 'last\_10'-dataset op de 'pid'-kolom, met behulp van de 'merge()' methode. Dit creëert een nieuwe dataset 'joined\_pr' die alleen de pull requests en bestanden bevat die in beide datasets voorkomen.

# Identify the unique files

files = set(joined\_pr['file'])

de 'joined\_pr'-dataset wordt opgeslagen bij de Files variabele komt door de set()'

# Print the results

Files

Deze zorgen dat de resultaten de naam “Files” meekrijgen.

https://app.datacamp.com

https://www.w3schools.com/python/pandas/default.asp

7

# This is the file we are interested in:

file = 'src/compiler/scala/reflect/reify/phases/Calculate.scala'

# Identify the pull requests that changed the file

file\_pr = data[data['file'] == file]

# Count the number of changes made by each developer

author\_counts = file\_pr.groupby('user').count()

# Print the top 3 developers

author\_counts.nlargest(3, 'file')

Deze code zoekt uit welke ontwikkelaars de meeste wijzigingen hebben aangebracht in een specifiek bestand. Eerst wordt er een variabele 'file' gedefinieerd die het pad naar het betreffende bestand bevat. Vervolgens worden alle pull requests uit een dataset 'data' geselecteerd die het betreffende bestand hebben gewijzigd en opgeslagen in een nieuwe variabele 'file\_pr'. Daarna worden de pull requests gegroepeerd op gebruikersnaam ('user') en het aantal wijzigingen per gebruiker geteld. Deze informatie wordt opgeslagen in een variabele 'author\_counts'. Uiteindelijk worden de top 3 ontwikkelaars ('user') geprint die de meeste wijzigingen hebben aangebracht in het betreffende bestand, gebaseerd op het aantal wijzigingen. Dit gebeurt door de 'nlargest()' methode te gebruiken op de 'author\_counts'-dataset en het aantal gewijzigde bestanden ('file') als het sorteerargument op te geven.

https://app.datacamp.com

https://www.w3schools.com/python/pandas/default

8

file = 'src/compiler/scala/reflect/reify/phases/Calculate.scala'

# Select the pull requests that changed the target file

file\_pr = pull\_files[pull\_files['file'] == file]

# Merge the obtained results with the pulls DataFrame

joined\_pr = pulls.merge(file\_pr, on='pid')

# Find the users of the last 10 most recent pull requests

users\_last\_10 = set(joined\_pr.nlargest(10, 'date')['user'])

# Printing the results

users\_last\_10

Deze code zoekt naar de gebruikers die de meest recente wijzigingen hebben aangebracht aan een specifiek bestand. Eerst wordt er een variabele 'file' gedefinieerd met het pad naar het betreffende bestand. Vervolgens worden alle pull requests uit een dataset 'pull\_files' geselecteerd die het betreffende bestand hebben gewijzigd en opgeslagen in een nieuwe variabele 'file\_pr'. Daarna worden de resultaten samengevoegd met een andere dataset 'pulls', op basis van het pull request ID ('pid'). Vervolgens worden de gebruikers geselecteerd die betrokken waren bij de 10 meest recente pull requests die het betreffende bestand hebben gewijzigd. Deze informatie wordt opgeslagen in een variabele 'users\_last\_10'. Ten slotte worden de gebruikers geprint naar de console.

https://app.datacamp.com

https://www.w3schools.com/python/pandas/default

9

%matplotlib inline

# The developers we are interested in

authors = ['xeno-by', 'soc']

# Get all the developers' pull requests

by\_author = pulls[pulls['user'].isin(authors)]

# Count the number of pull requests submitted each year

counts = by\_author.groupby([by\_author['user'], by\_author['date'].dt.year]).agg({'pid': 'count'}).reset\_index()

# Convert the table to a wide format

counts\_wide = counts.pivot\_table(index='date', columns='user', values='pid', fill\_value=0)

# Plot the results

counts\_wide.plot(kind='bar')

10

authors = ['xeno-by', 'soc']

file = 'src/compiler/scala/reflect/reify/phases/Calculate.scala'

# Merge DataFrames and select the pull requests by the author

by\_author = data[data['user'].isin(authors)]

# Select the pull requests that affect the file

by\_file = by\_author[by\_author['file'] == file]

# Group and count the number of PRs done by each user each year

grouped = by\_file.groupby(['user', by\_file['date'].dt.year]).count()['pid'].reset\_index()

# Transform the data into a wide format

by\_file\_wide = grouped.pivot\_table(index='date', columns='user', values='pid', fill\_value=0)

# Plot the results

by\_file\_wide.plot(kind='bar')

Op github zie je aan de linkerkant al je repositories en daar kan je de juiste aan klikken. Als je die open hebt geklikt dan kan je geuploade documenten zien en als je die openklikt kan je ze online bewerken of donwloaden naar je PC.