cphbusi

COPENHAGEN BUSINESS ACADEMY





Classification and geospatial analysis

Jens Egholm Pedersen
<jeep@cphbusiness.dk>

Recap

- Populations and samples
- Distributions
- Regression functions
- Linear regression
  - Multivariate regression
- Polynomial regression
- Fitting
- Vectors, matrices and tensors
- Dimensionality and dimension reduction

cphbusiness

Hand-in 5

- Good work
- Linear regression
- MAE vs. MSE vs. Pearson’s r
- Answering what you’re being asked

Guest lecturer + next week

- We invited a guest lecturer
  - Still happening next Tuesday

- .. But I'm travelling!
  - Suggestion: final lecture on Friday 1st

Goal of this block

- Have a basic understanding and knowledge of various terms, models and tests in statistics.
- Compute basic statistics on data using the Python's scientific stack and the Sklearn library.
- Develop an informed guess of when to choose a certain model to answer a concrete type of question and apply technology appropriately.

See also: BI plan

Goal for today

- Hand-in debriefing
- Gradient descent
- Clustering
- Classification
  - Text analysis
- Geospatial data analysis

See also: BI plan

Types of machine learning

- Is it trained by a human
  - Supervised / unsupervised
- Can they learn on the fly?
  - Online learning / offline (batch) learning
- Do they include new data?
  - Instance-based / model-based
- Today: supervised, offline model-learning
  - And unsupervised, instance based!

See also: Géron: Hands-on machine learning (book)

Model optimisation

- How did we get our linear model?

- Let's have a look at the model

  $y = ax + b$

  $E(y) = ax+b$

- How would you optimise this?

See also: Gradient descent on Wikipedia

Gradient descent

- Let's have a look at the model

  $y = ax + b$

  $E(y) = ax+b$

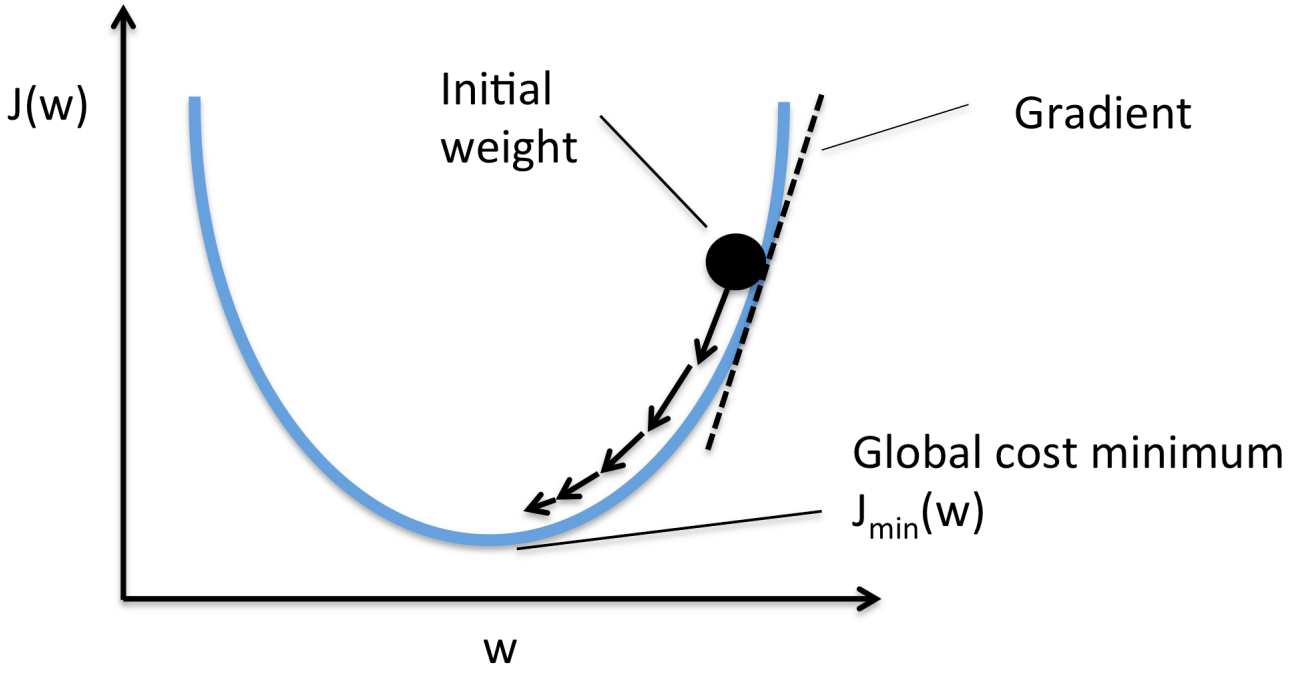
- How would you optimise this?

- Least squares method
  - scipy.linalg.lstsq
  - Used in your Linear Regression
- Naive approach: Try many different a and b
  - If the error value is better
  - = gradient descent

See also: Gradient descent on Wikipedia, scipy.linalg.lstsq

Gradient descent

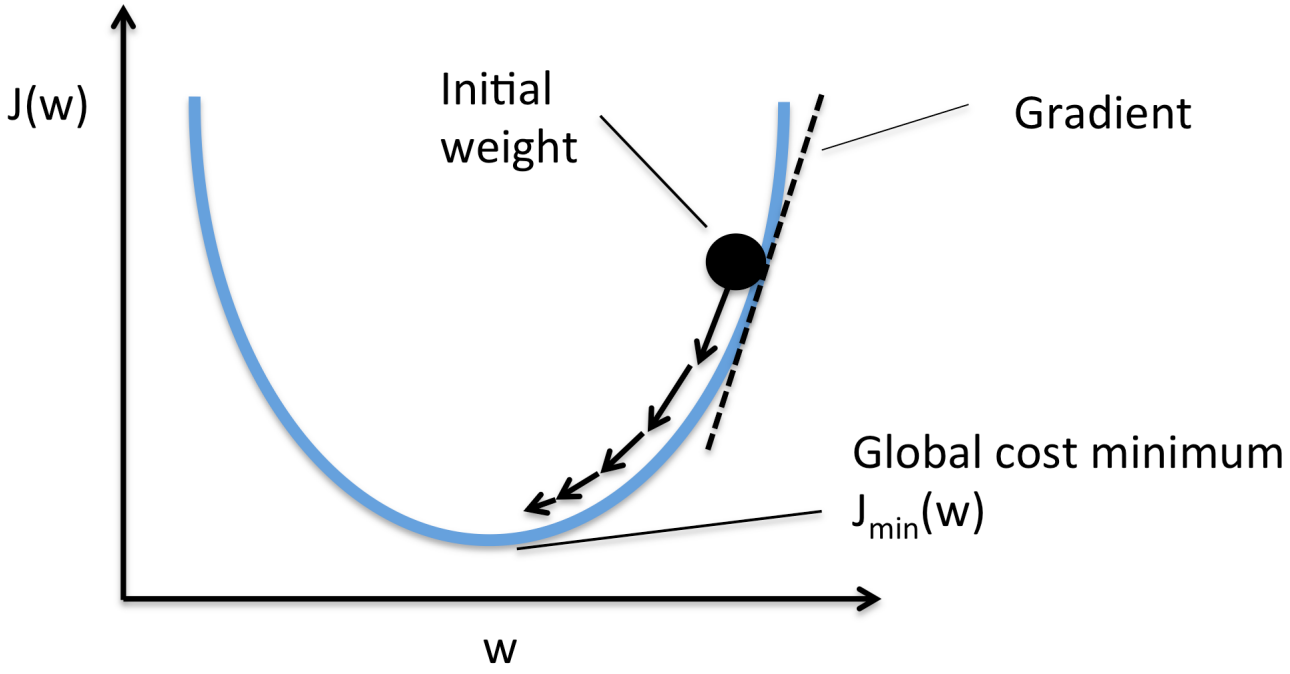
- E(y) = ax+b



See also: Regression assignment, Brown university

Gradient descent

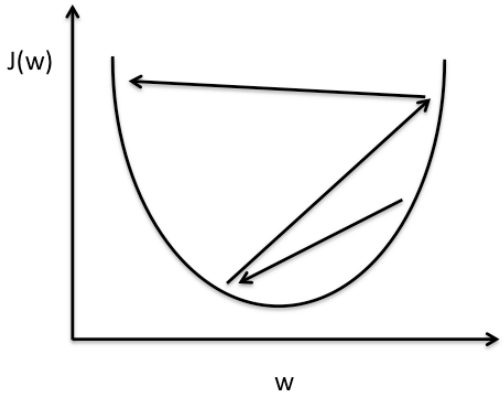
- E(y) = ax+b
  - If a/b are too small: bigger errors
  - If a/b are too big: bigger errors



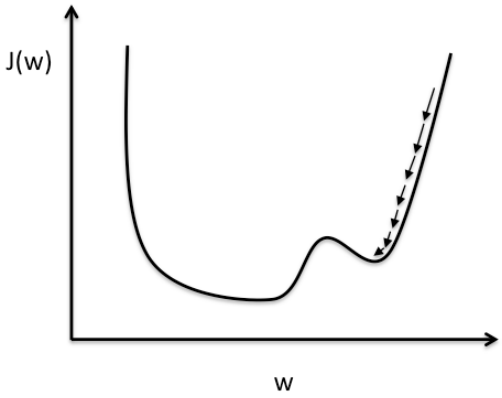
See also: Regression assignment, Brown university

Gradient descent

- E(y) = ax+b
- Problem: Local minima



Large learning rate: Overshooting.

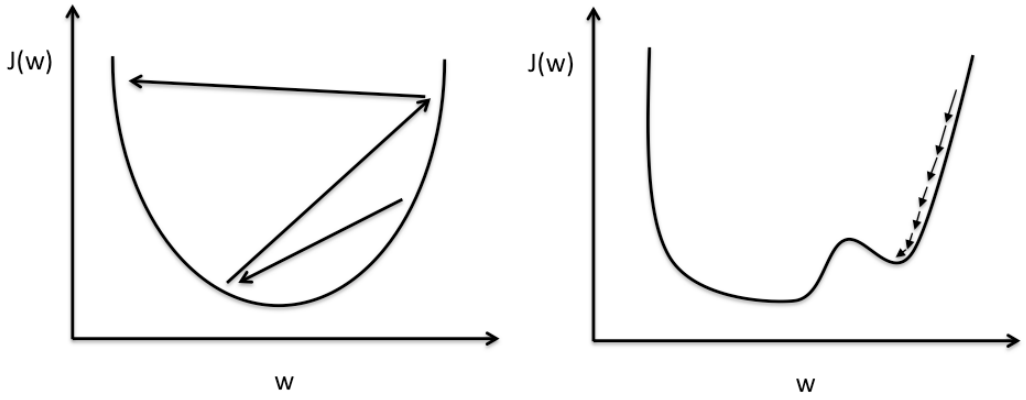


Small learning rate: Many iterations until convergence and trapping in local minima.

See also: Regression assignment, Brown university

Stochastic gradient descent

- E(y) = ax+b
- Problem: Local minima
- Solution: Randomise “jump” distance
  - Trade-off: Overshooting versus local minima



Large learning rate: Overshooting.

Small learning rate: Many iterations until convergence and trapping in local minima.

See also: Regression assignment, Brown university

More about vectors

- Vectors are just arrays
  - One number in the array represent one dimension
  - [0, 0]
  - [1, 1]
  - [-2, 2]
  - [0, 0, 0]
  - [1, 1, 1]
  - [-2, 2, 3

More about vectors

- Vectors are just arrays

- What if the vector has more than three numbers?
  - Hard to visualise
  - Maybe hard to model?

See also: sklearn on clustering

More about classification

- What if you don’t have a clear answer?

- Supervised learning
  - Input data (X)
  - Training and testing with predicting data (y)

- Unsupervised learning
  - Input data (X)
  - Is there a pattern?

See also: sklearn on clustering

Recognising faces

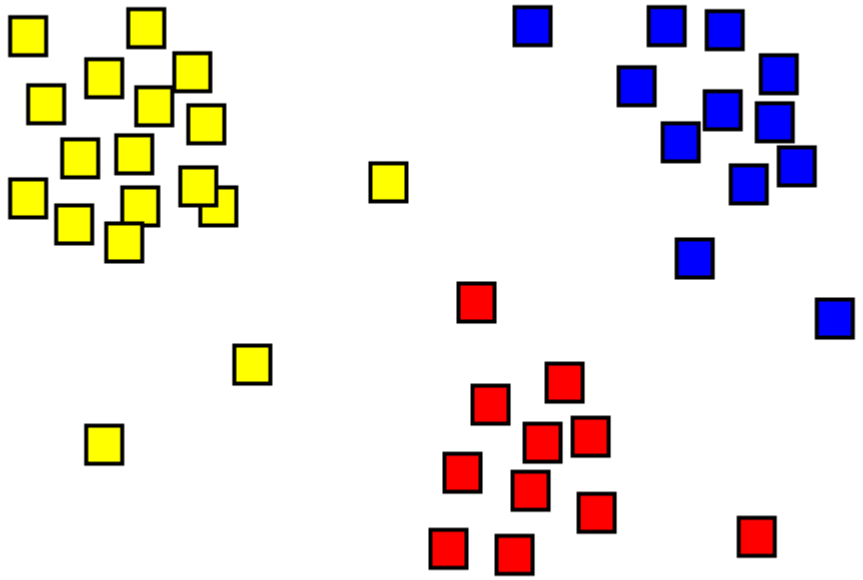
- What defines an image?

- Can you think of the pixels in a different way?

See also: Face recognition

Recognising faces

- Take an array of pixels
  - Flatten them, so each pixel value is one dimension
- Clustering
  - “Groups” of numbers



See also: Face recognition

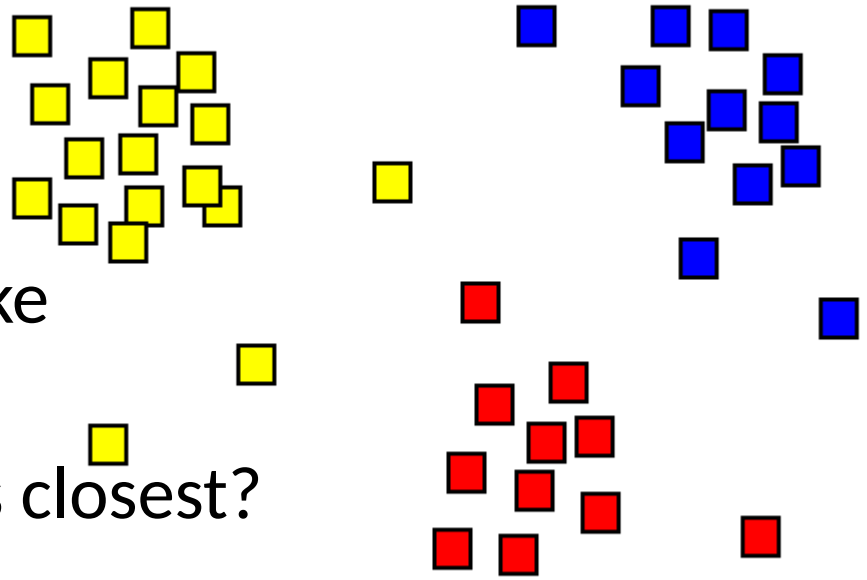
Cluster analysis

1) Take a list of points in (high-dimensional) space

2) Plot them

3) See how many
   "groups" you can make

4) Predicting == which is closest?



See also: Face recognition

Clustering

- "Grouping" into *n* classes
  - K-means clustering

```
from sklearn.cluster import Kmeans
kmeans = KMeans(n_clusters=6)
kmeans.fit(X)
kmeans.cluster_centers_
```

See also: sklearn on clustering

Recap

- Gradient descent
- Clustering
- Classification
  - Text analysis
- Geospatial data analysis

See also: BI plan

Text analysis

- What is text?
  - Letters? Books? DNA?
- `String : [Char]`
- For us:
  - Sequence of letters containing semantic meaning

See also: String on Wikipedia

NLTK

- Natural Language ToolKit (NLTK) in Python

- Seriously cool toolkit
  - Text parsing
  - Machine translation
  - Semantic parsing
  - Sentiment analysis
  - Etc.

- We will use semantic analysis only

See also: http://www.nltk.org/

Semantics

- Linguistic and philosophical study of meaning

- Primarily concerned with *relationships*

- Happy vs sad
- Angry vs satisfied
- Etc…

See also: BI plan

nltk.sentiment.vader

- Sentiment analysis tool
  - Hutto, C.J. & Gilbert, E.E. (2014). VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text. Eighth International Conference on Weblogs and Social Media (ICWSM-14). Ann Arbor, MI, June 2014.

- nltk.sentiment.vader

See also: BI plan

nltk.sentiment.vader

```
import nltk.sentiment.vader
nltk.download('vader_lexicon')

model = SemtimentIntensityAnalyzer()
model.polarity_scores()
```

See also: BI plan

Text analysis

- What is happy? Or sad?
- What is meaning?
- How do we “measure” words?
- Sentiment analysis helps us to “rate” words on predefined dimensions
  - Based on a pre-known dictionary

See also: String on Wikipedia

Text analysis

- Sentiment analysis helps us to “rate” words on predefined dimensions
  - Based on a pre-known dictionary

- What if we let the machine decide?

See also: String on Wikipedia

Word/term frequency

- “I’m very very very happy”
- “Ice cream, ice cream everybody wants ice cream”

- Term frequency
  - The number of times a word occurs in a document

See also: String on Wikipedia

Word weight

- “I’m very very very happy”
- “I’m happy”
- How much information does “happy” provide?
- Inverse document frequency
  - Is the word common across all documents?
  - Measures how much information is in a word

See also: String on Wikipedia

TF/IDF

- Term frequency
  - The number of times a word occurs in a document
- Inverse document frequency
  - Is the word common across all documents?
  - Measures how much information is in a word
- TF/IDF
  - Reflects *how important* a word is in the document
  - Now only for words

See also: String on Wikipedia

TF/IDF

- Term frequency
  - The number of times a word occurs in a document
- Inverse document frequency
  - Is the word common across all documents?
  - Measures how much information is in a word
- TF/IDF
  - Reflects *how important* a word is in the document
  - Not only for words

See also: String on Wikipedia

TF/IDF in sklearn

```
from sklearn.feature_extraction.text import
TfidfVectorizer


model = TfidfVectorizer()

model.fit(<list of text>)

model.transform(<list of text>)


model.fit_transform(<list of text>)
```

See also: String on Wikipedia

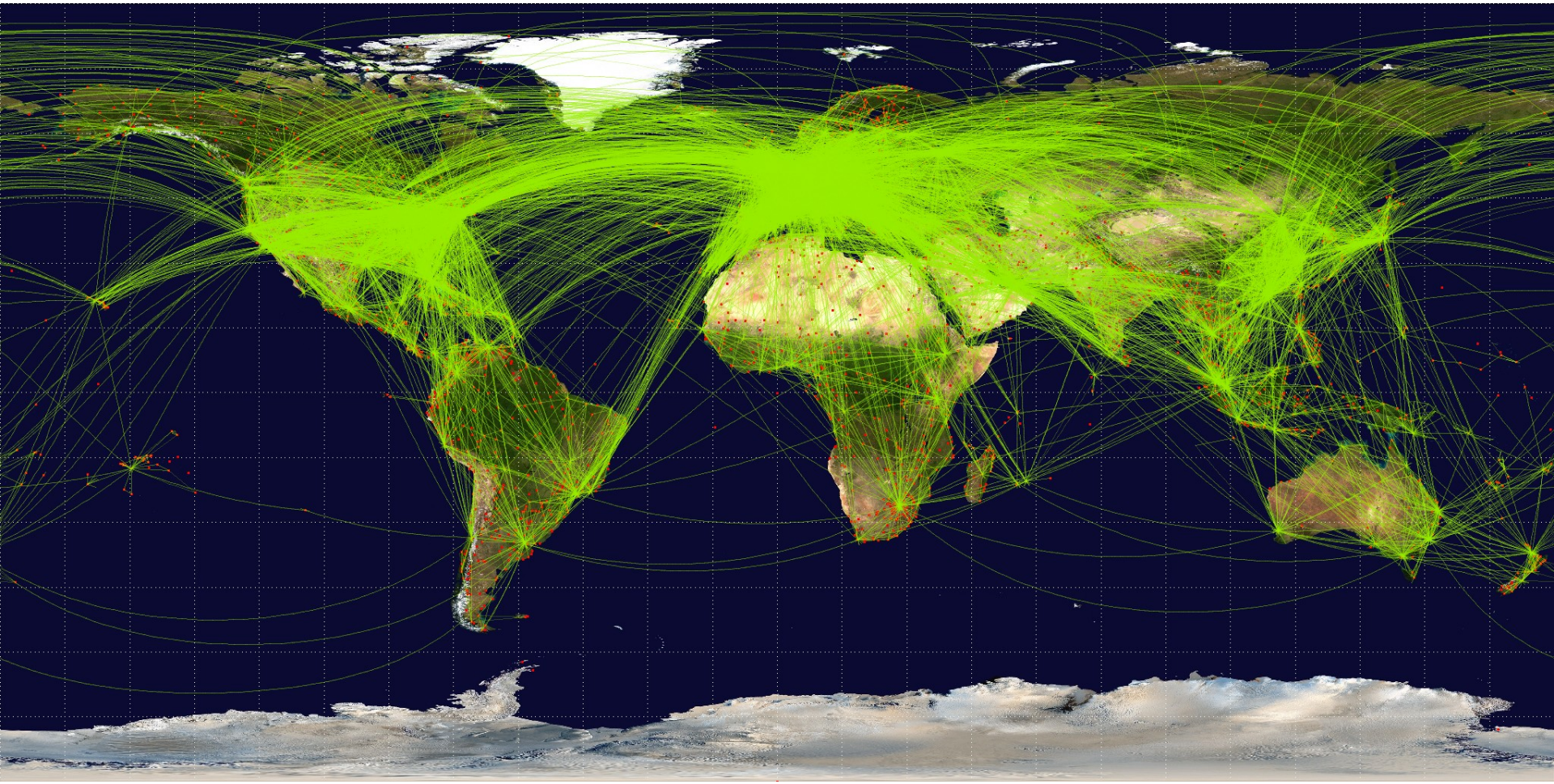
Recap

- Gradient descent
- Clustering
- Classification
  - Text analysis
- Geospatial data analysis

See also: BI plan

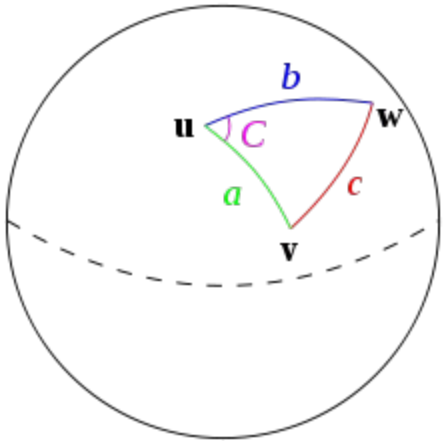
cphbusiness

Geospatial data



Geospatial data

- Basically a coordinate (x, y)
- … But the earth is a sphere
  - Airplanes don’t fly straight lines
  - Haversine distance metric



- What kinds of analysis can we do with them?
  - Finding close and cheap pizzas!

See also: Haversine formula

Geospatial information

- Folium

- Map
  - `import folium`
  - `m = folium.Map(location=[x, y])`
- Markers
  - `folium.Marker([x, y], popup=”Hullu Bullu”).add_to(m)`
- Heatmap
  - `from folium.plugins import HeatMap`
  - `HeatMad(data).add_to(m)`

See al\so: Folium documentation

Next hand-in: Assignment 7

- Deadline: **26th of November 23:59:59**
- Text classification using VADER sentiment analysis and KMeans neighbouring cluster
- Geospatial analysis and prediction of best place to buy housing

Next hand-in: Assignment 7

- Deadline: **26th of November 23:59:59**
- The hand-in (on Moodle) should be a link to a GitHub release containing a single file with the code and written text for the assignment parts

- This can either be a .ipynb, .py, .pdf or .md file
- The file must be clearly identifiable. Please name it accordingly. (for instance report.pdf)