Bolighed - powered by Python

A real life case study

About me

- Mathematician by education
- After some years in research I have since worked as a Python developer, primarily in data processing at:
    - Danish Geodata Agency, now called SDFE
    - Danish Meteorological Institute
    - And now, **Bolighed A/S**

About Bolighed

- Bolighed is a website aimed at house owners and hunters as well.
- Collects and presents information from a lot of public data sources:
  - BBR (basic information about buildings, dwellings and ownership).
  - Tinglysning (loans, entitlements etc.)
  - Energy marks
  - …
- Also a lot of “closed” non-public sources:
  - Price estimate models (machine learning)
  - Sales data
  - ...

The stack at Bolighed

Advanced setup with a *lot* of components:

- Amazon EC2
- Docker
- Redis
- Eleasticsearch
- Cloudflare
- Postgres / Postgis (databases)
- Nginx
- Tornado (to be phased out…)
- … and a whole lot more ...

Where is Python used?

The frontend is using AngularJS (soon - preact) and Python takes care of the rest:

- Data import
- Infrastructure
  - Deployment / configuration via **ansible**
- Backend api
  - Flask
  - Django
- Data analysis
  - Numpy, scipy, pandas, matplotlib, scikit-learn, SQL via SQLAlchemy.

A deeper look into some of the use cases

Backend:

- Flask with SQLAlchemy
- Super simple and very flexible setup:

```python
from flask import Flask
app = Flask(__name__)

@app.route('/')
def hello_world():
    return 'Hello, World!'
```

Backend

Why Python and not PHP, C, C# or Java (or Ruby)? Is performance OK??

- Much, much nicer and more maintainable than PHP!
- Very high level interface to various services / infrastructure
  - Elasticsearch, Redis, Postgres (SQLAlchemy), Datadog, Amazon EC2 / S3.
- Lot’s of caching mechanisms and load balancing in place - very few actual database calls...

Backend

We also have some api's running in Django:

- More structured than Flask + SQLAlchemy
- Includes it's own ORM (Object-relational mapping) as a high level interface to the database.
- Lot's of extensions, e.g.
- Used by many *huge* web applications out there:
    - **Instagram**
    - **Pinterest**
    - ...

Django’s ORM

```python
class CustomerType(models.Model):
    created = models.DateTimeField(auto_now_add=True)
    modified = models.DateTimeField(auto_now=True)
    name = models.CharField(max_length=255, unique=True)
    def __str__(self):
        return self.name


class PropertyData(models.Model):
    """
    Models any kind of property
    """
    bbr_property_data = models.ForeignKey('BBRPropertyData', null=True)
    address = models.ForeignKey('Address', null=True)
```

Django’s ORM

```
(venv_bm) Simons-MacBook-Pro:business_manager simonkokkendorff$ python manage.py shell
Python 3.6.0 (default, Dec 24 2016, 08:01:42)
Type "copyright", "credits" or "license" for more information.
...
In [1]: from business_manager.leads import models
 In [2]: for obj in models.Address.objects.all().filter(street__startswith="Åsvej")[:2]:
   ...:     print(obj)
   ...:
Åsvejen 4  , 4330
Åsvejen 6  , 4330
```

- Specific database is ‘abstracted away’
- No explicit SQL queries
- However, in some cases the high level ORM is too rigid and one must resolve to plain old SQL...

Data import

We use a lot of different python libraries and protocols for fetching data from various sources:

- **Boto / boto3** for talking to Amazon EC2 and S3
- **Requests** for REST-interfaces / scraping
- **Pysimplesoap / Requests** for SOAP (XML) interfaces (sigh….)

For example there is a great API for all danish addresses at http://dawa.aws.dk/

```
In [14]: import requests
In [15]: r = requests.get("http://dawa.aws.dk/adresser", params={"vejnavn":"Fasanvej", "postnr": 8210, "husnr":15, "struktur":"mini"})
In [16]: r.json()
Out[16]:
[{'adgangsadresseid': '0a3f5096-212e-32b8-e044-0003ba298018',
  'dør': None,
  'etage': None,
  'husnr': '15',
  'id': '19910d90-1d47-41c9-e044-0003ba298018',
  'kommunekode': '0751',
  'postnr': '8210',
  'postnrnavn': 'Aarhus V',
  'status': 1,
  'supplerendebynavn': None,
  'vejkode': '2032',
  'vejnavn': 'Fasanvej',
  'x': 10.1787079932534,
  'y': 56.1647588529531}]
```

Addresses, postal districts and various other data are imported from this endpoint on a regular basis.

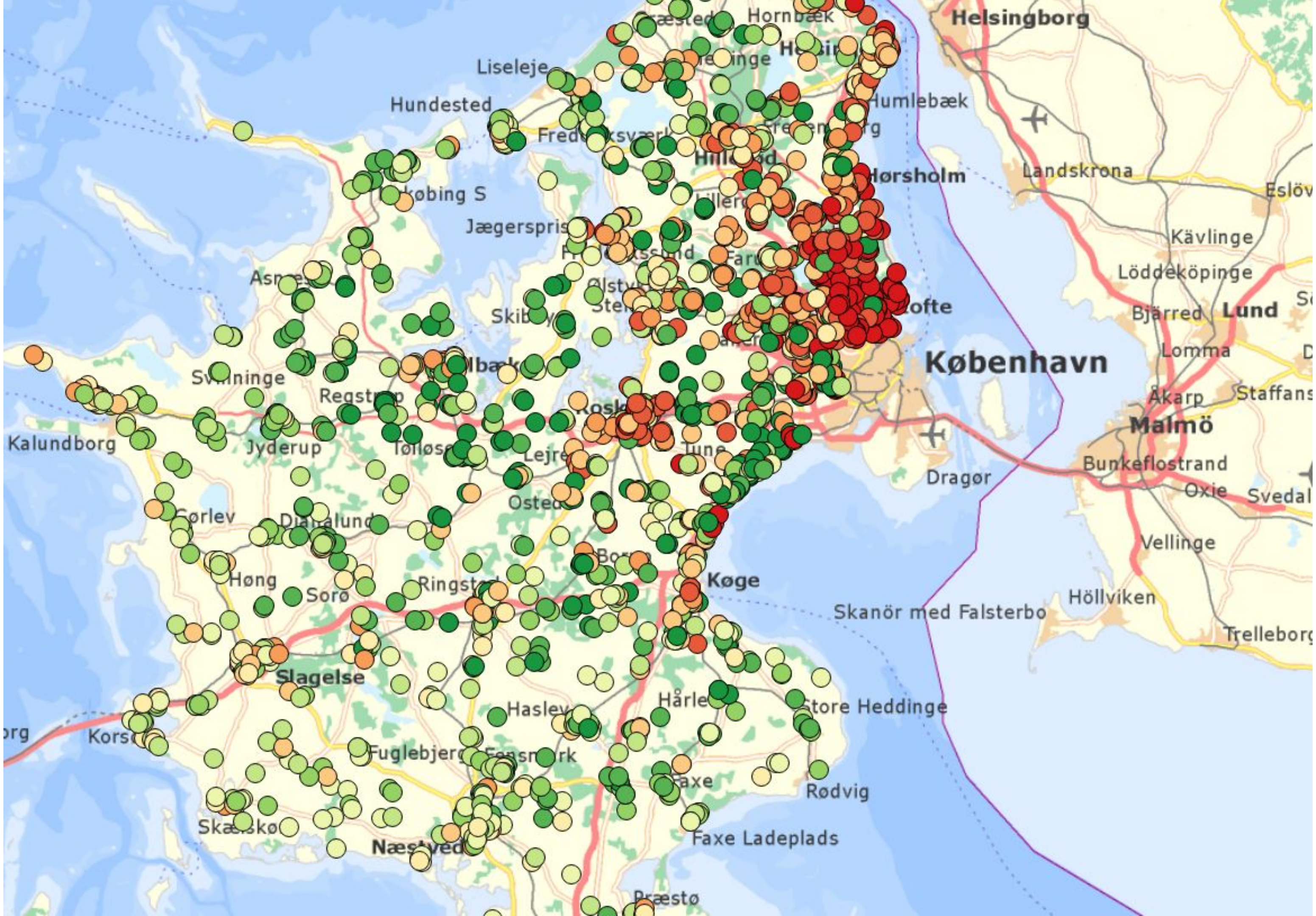
Data analysis

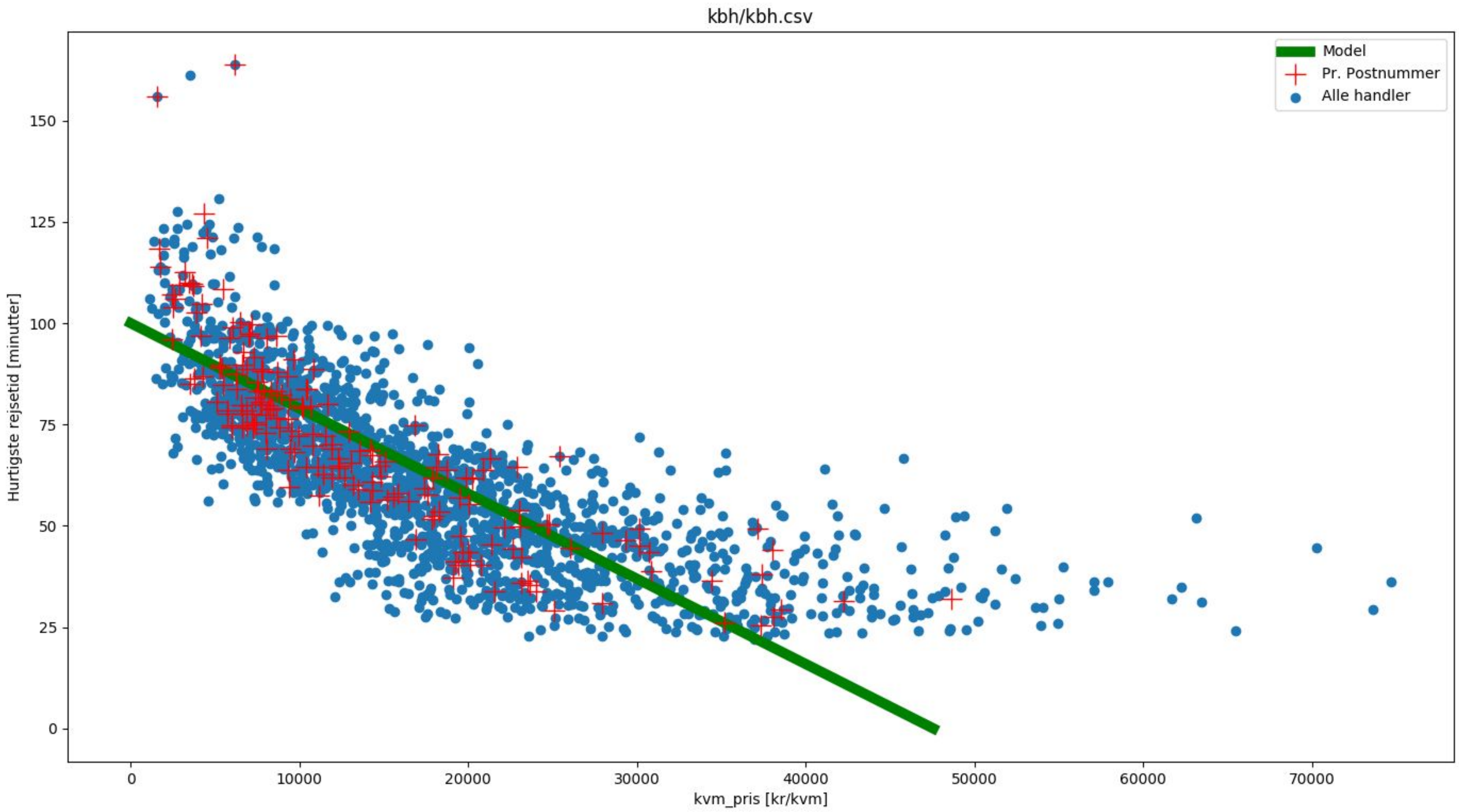
Case:

- Examine the relation between house prices and travel time to Copenhagen.

Plan:

- Fetch sales data + geographic location from database (Postgis) via SQLAlchemy.
- Use googlemaps Python API to query travel times to Copenhagen Central station for these locations.
- Do some analysis and plotting with numpy (linear regression, filtering) and matplotlib


Helsingborg
Hornbæk
Liseleje
Hundested
Humlebæk
Hørsholm
Landskrona
Kävlinge
Löddeköpinge
Bjärred
Lund
Lomma
København
Åkarp
Malmö
Bunkeflostrand
Oxie
Svedala
Dragør
Vellinge
Höllviken
Skanör med Falsterbo
Køge
Store Heddinge
Rødvig
Faxe Ladeplads
Præstø
Slagelse
Kalundborg
Jyderup
Regstrup
Sorø
Høng
Haslev
Hårlev
Fuglebjerg
Osted
Lejre
Tune
Jægerspris



kbh/kbh.csv
Model
Pr. Postnummer
Alle handler
Hurtigste rejsetid [minutter]
0
25
50
75
100
125
150
0
10000
20000
30000
40000
50000
60000
70000
kvm_pris [kr/kvm]

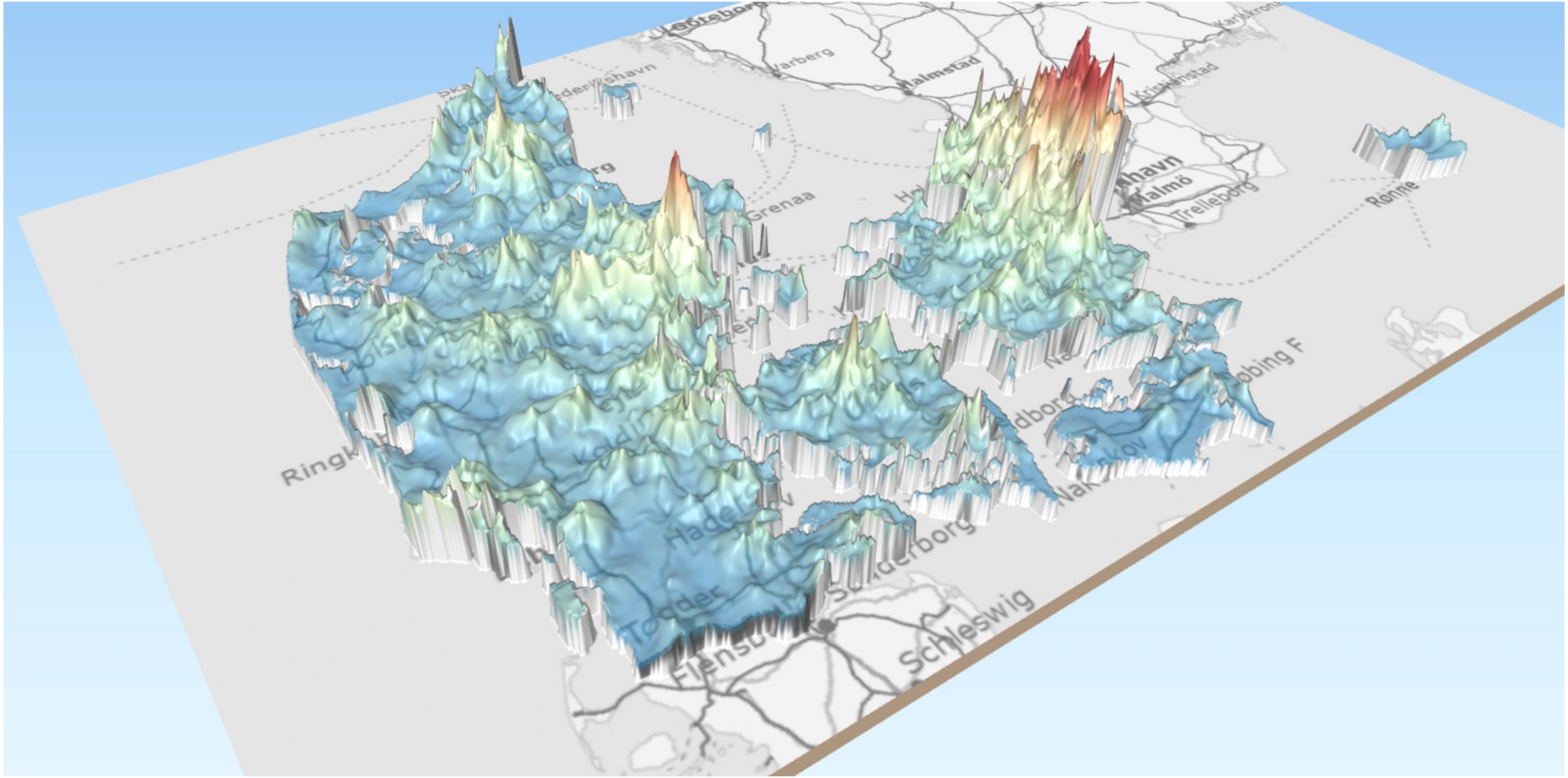
Something else that I’ve been working on...

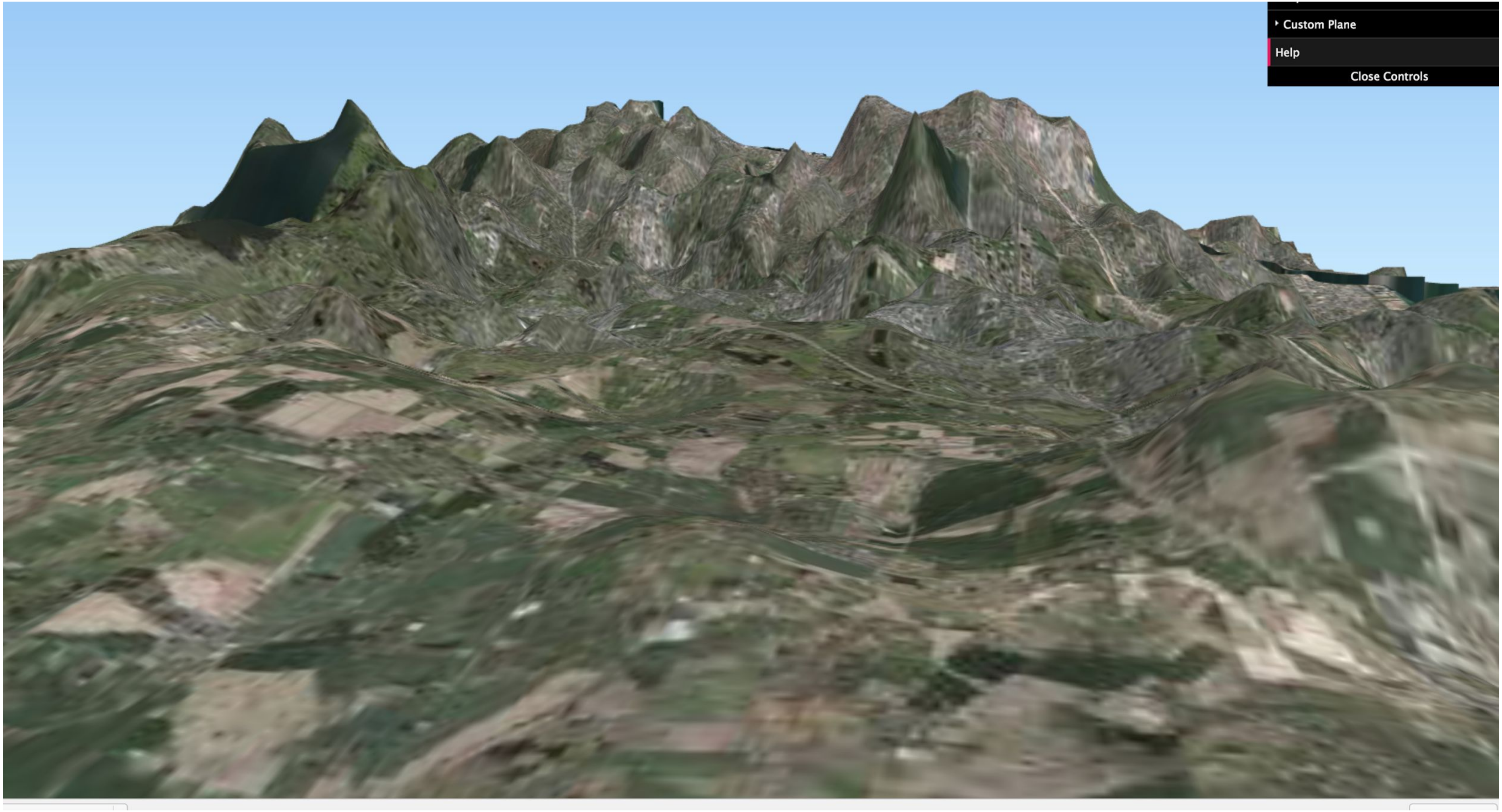
- Mapping value increases for houses the next year:
    - https://s3.bolighed.dk/static/stories/prisprognose/index.html#7/56.188/11.646
- And something completely different - a fancy map:
    - http://gittebach.dk/case/story.html
- How does house prices depend on various parameters?
    - For example energy marks?
    - Create models using scikit-learn…
    - ...or tensorflow … or...

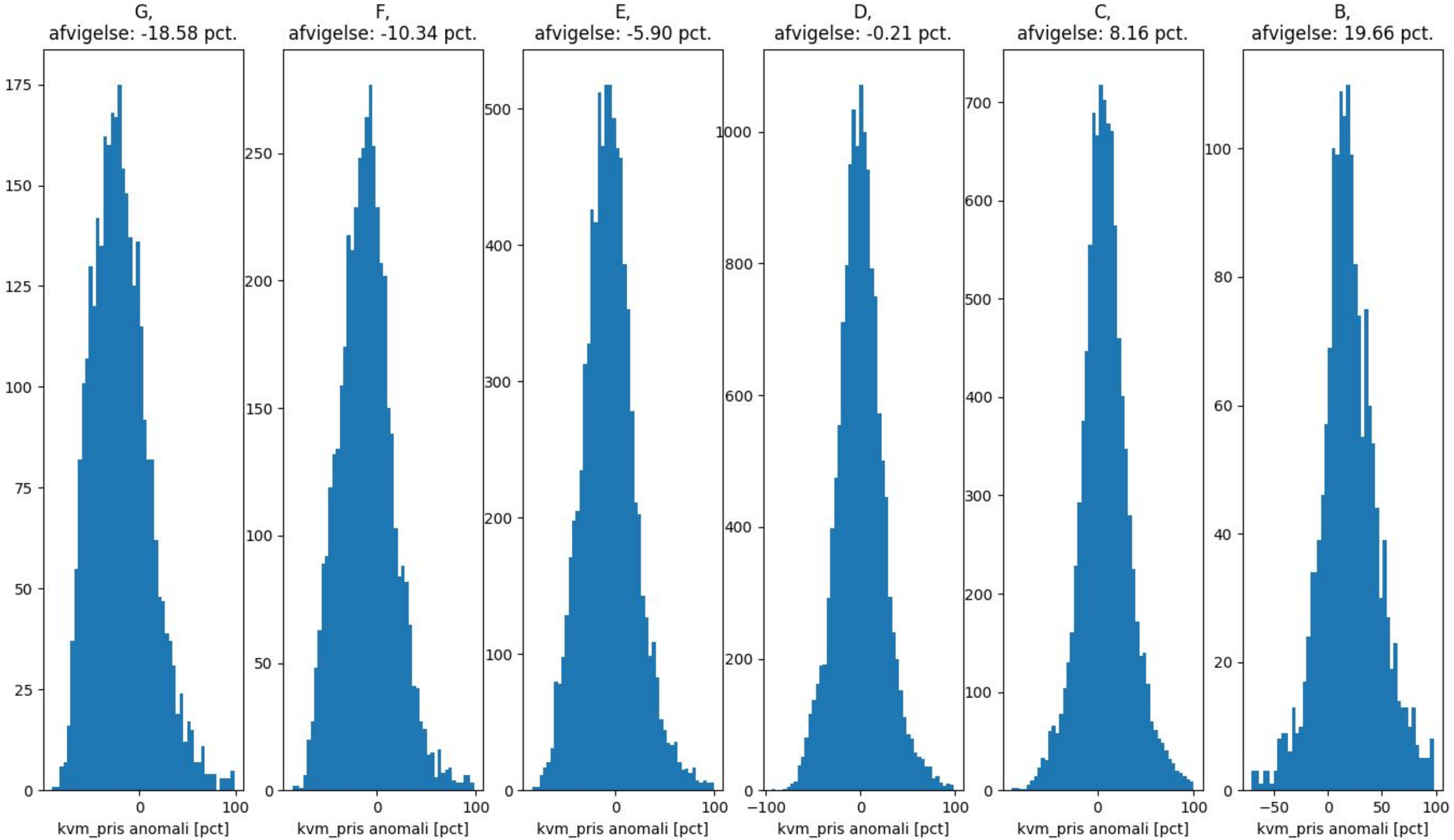
Work in progress - analysis of price versus energy mark

Try to analyse if energy mark has a significant influence on sales price?

- Prices are very different across the country, so don’t do this in an absolute scale (kr.)
- We know that location is actually the most important parameter for the price of a house (location, location, location…), so try to factor this out somehow.
- Hmm, make a price model and look at deviations from this instead.


Varberg
Halmstad
Karlskrona
Grenaa
Malmö
Trelleborg
Rønne
Schleswig


Custom Plane
Help
Close Controls


G,
afvigelse: -18.58 pct.
F,
afvigelse: -10.34 pct.
E,
afvigelse: -5.90 pct.
D,
afvigelse: -0.21 pct.
C,
afvigelse: 8.16 pct.
B,
afvigelse: 19.66 pct.
kvm_pris anomali [pct]

Linear regression analysis with statsmodels

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  anoma   R-squared:                       0.100
Model:                            OLS   Adj. R-squared:                  0.100
Method:                 Least Squares   F-statistic:                     952.8
Date:                Wed, 21 Jun 2017   Prob (F-statistic):               0.00
Time:                        14:43:57   Log-Likelihood:            -2.0021e+05
No. Observations:               42858   AIC:                         4.004e+05
Df Residuals:                   42852   BIC:                         4.005e+05
Df Model:                           5
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept     19.6552      0.628     31.320      0.000      18.425      20.885
C(em)[T.2]   -11.4971      0.678    -16.945      0.000     -12.827     -10.167
C(em)[T.3]   -19.8643      0.663    -29.973      0.000     -21.163     -18.565
C(em)[T.4]   -25.5576      0.687    -37.187      0.000     -26.905     -24.211
C(em)[T.5]   -29.9923      0.734    -40.846      0.000     -31.431     -28.553
C(em)[T.6]   -38.2364      0.775    -49.366      0.000     -39.755     -36.718
==============================================================================
Omnibus:                     1279.801   Durbin-Watson:                   1.984
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             1767.298
Skew:                           0.330   Prob(JB):                         0.00
Kurtosis:                       3.745   Cond. No.                         14.0
==============================================================================
```

Price “anomaly” versus energy mark.
Significant dependency?
Yes - but only explains a small part of the variation.

Thank you for your attention!

Some links:

- https://boligned.dk/
- https://da-dk.facebook.com/boligned/
- https://twitter.com/boligned