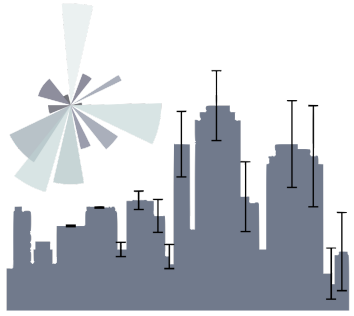
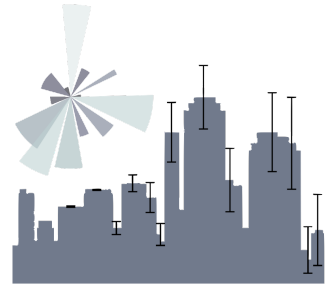
principles of Urban Science 5



spatial analysis

dr.federica bianco | *fbb.space* | *fedhere* | *fedhere*

this slide deck: https://slides.com/federicabianco/pus2020_5



1 geopandas

geopandas: GIS extension of pandas python package

GeoSeries and GeoDataFrame extend the functionality of the pandas data structures.

-> inherits all the functionality of pandas

-> added functionality for spatial analysis

builds on other python spatial packages to perform manipulation and analysis:

- fiona - coordinate
- shapely - shapes
- pyproj - projection
- rtree - for efficient cross operation
- pysal - spatial analysis

goepandas: GIS extension of pandas python package

GeoSeries and GeoDataFrame extend the functionality of the pandas data structures.

needs a geometry column

```
world.head()
```

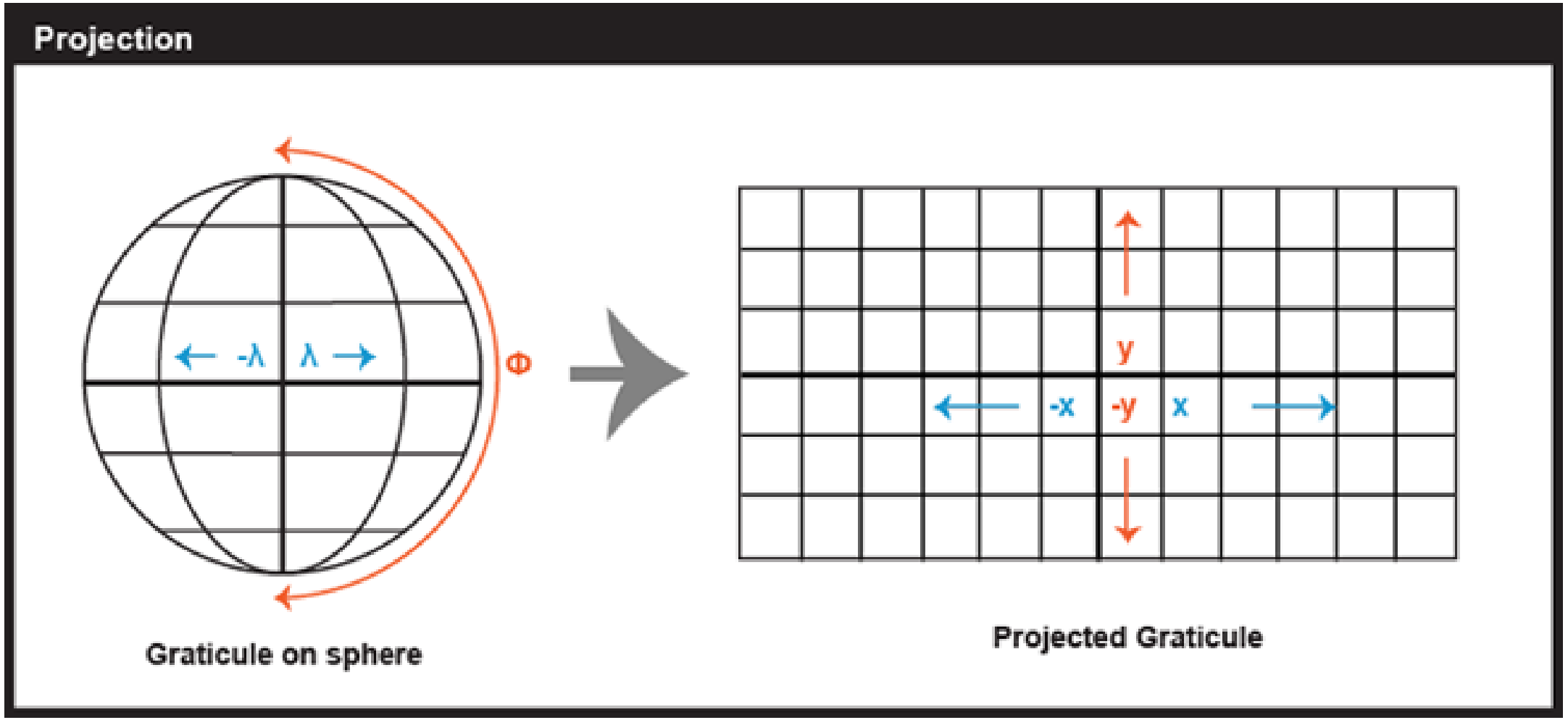
	pop_est	continent	name	iso_a3	gdp_md_est	geometry
0	920938	Oceania	Fiji	FJI	8374.0	MULTIPOLYGON (((23038880.464 -19387146.959, 22...
1	53950935	Africa	Tanzania	TZA	150600.0	POLYGON ((-29182486.766 -9001104.258, -2911968...
2	603253	Africa	W. Sahara	ESH	906.5	POLYGON ((-42268840.017 50089343.438, -4229576...

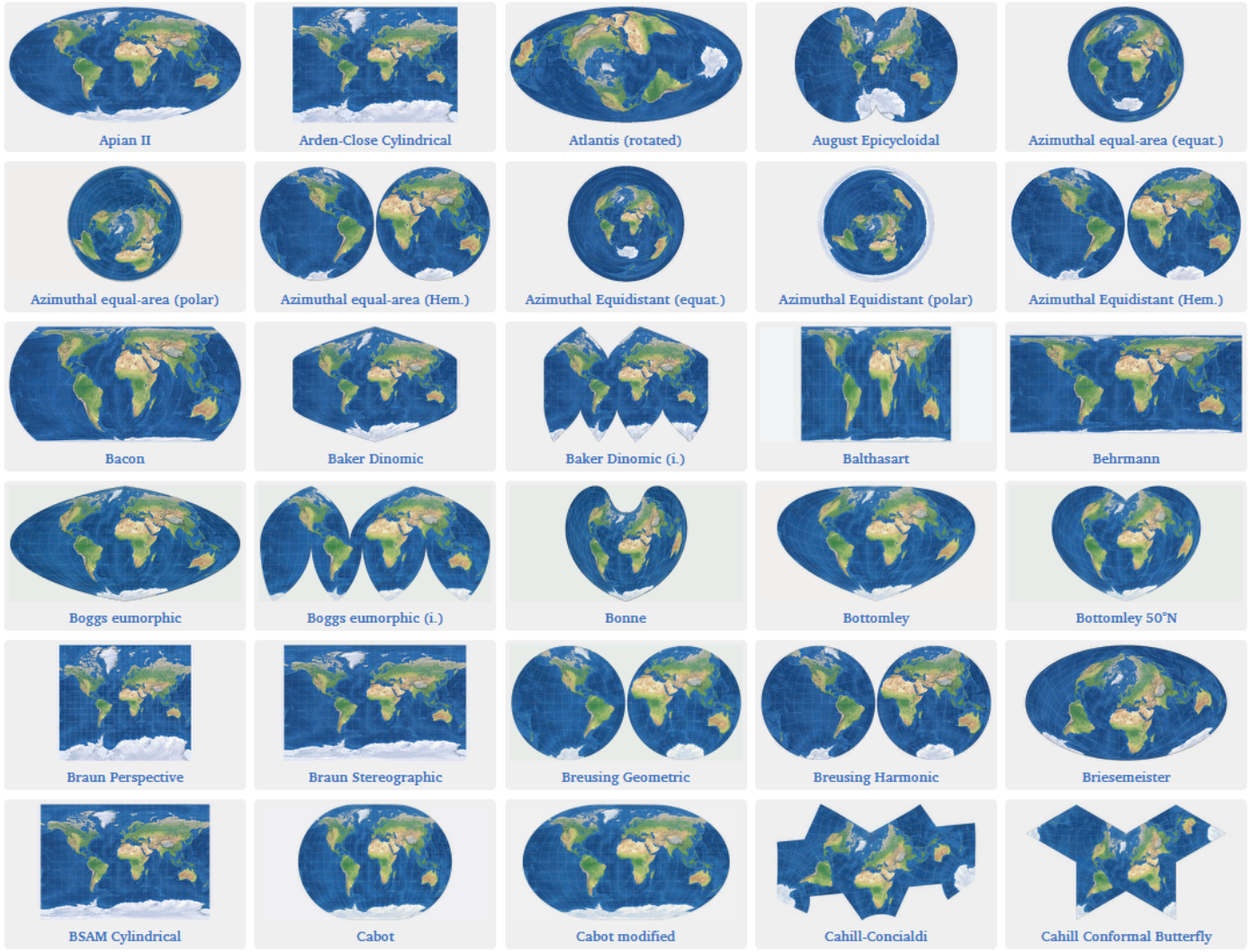
may contain coordinates in a separate attribute

```
world.crs
```

```
<Geographic 2D CRS: EPSG:4326>
Name: WGS 84
Axis Info [ellipsoidal]:
- Lat[north]: Geodetic latitude (degree)
- Lon[east]: Geodetic longitude (degree)
Area of Use:
- name: World
- bounds: (-180.0, -90.0, 180.0, 90.0)
Datum: World Geodetic System 1984
- Ellipsoid: WGS 84
- Prime Meridian: Greenwich
```

The earth is rounds... my monitor is flat





incomplete list of map projections

goepandas: GIS extension of pandas python package

GeoSeries and GeoDataFrame extend the functionality of the pandas data structures.

needs a geometry column

```
world.head()
```

	pop_est	continent	name	iso_a3	gdp_md_est	geometry
0	920938	Oceania	Fiji	FJI	8374.0	MULTIPOLYGON (((23038880.464 -19387146.959, 22...
1	53950935	Africa	Tanzania	TZA	150600.0	POLYGON ((-29182486.766 -9001104.258, -2911968...
2	603253	Africa	W. Sahara	ESH	906.5	POLYGON ((-42268840.017 50089343.438, -4229576...

the geometry column contains "shapely" objects

Shapely supports the creation of primitive geometry features. These include:

- Point
- LineString
- LinearRing
- Polygon
- Multipart

collections are also supported:

- MultiPoint,
- MultiLineString
- MultiPolygon

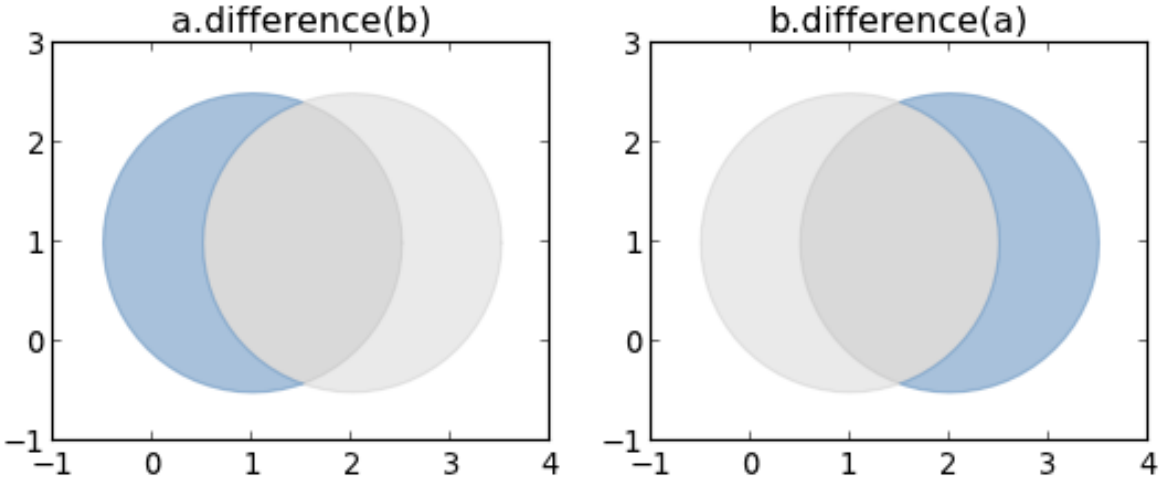
http://kjordahl.github.io/SciPy-Tutorial-2015/#1

Shapely

Shapely is a python library for geometric operations using the GEOS library.

Shapely can perform:

- geometry validation
- geometry creation (e.g. collections)
- geometry operations



http://kjordahl.github.io/SciPy-Tutorial-2015/#1

Shapely

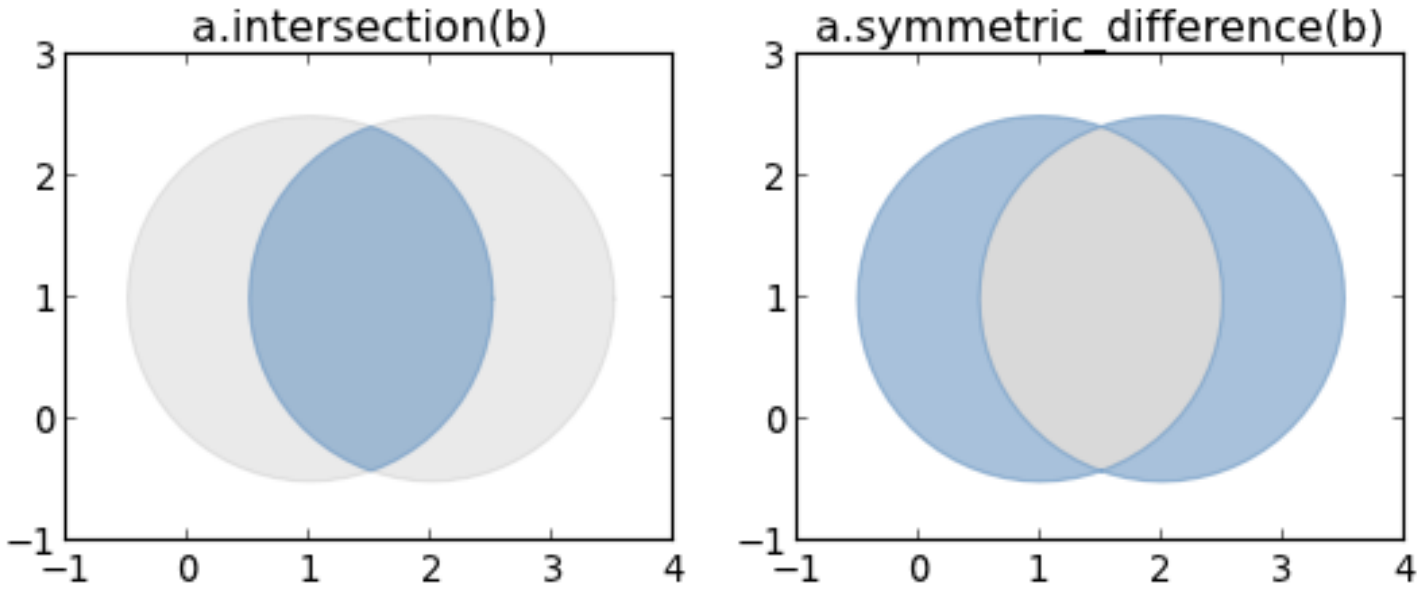
Shapely is a python library for geometric operations using the GEOS library.

Shapely can perform:

- geometry validation
- geometry creation (e.g. collections)
- geometry operations

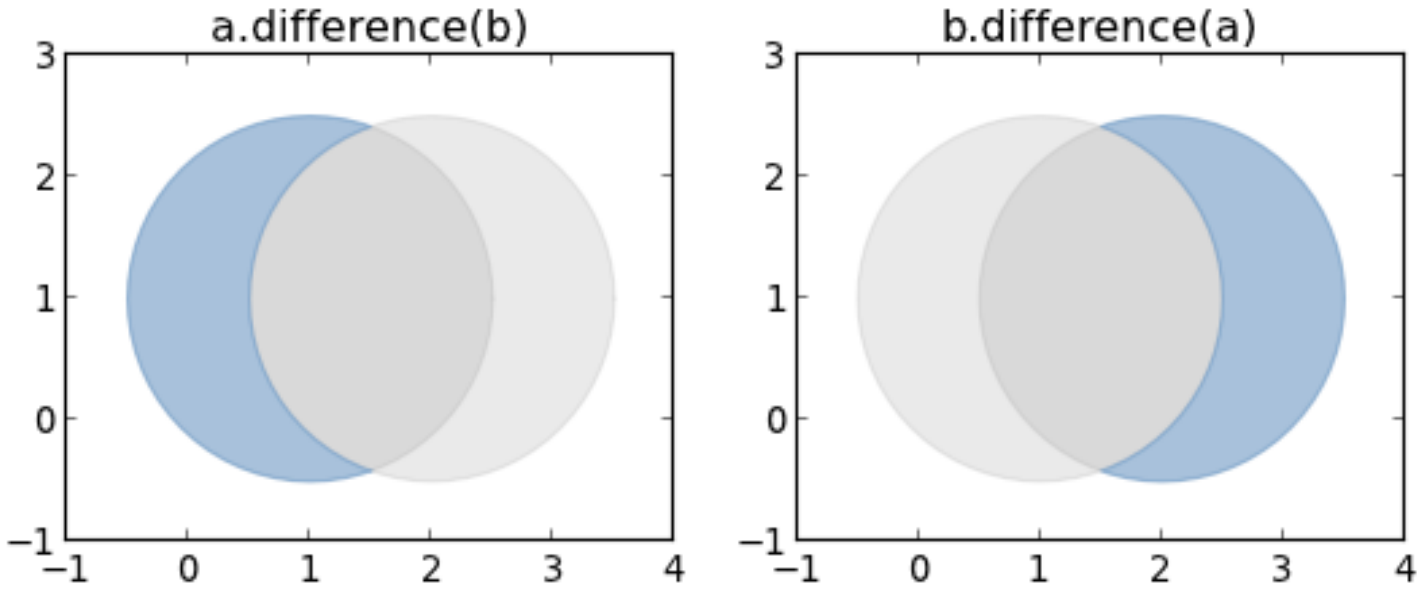
http://kjordahl.github.io/SciPy-Tutorial-2015/#1

Shapely



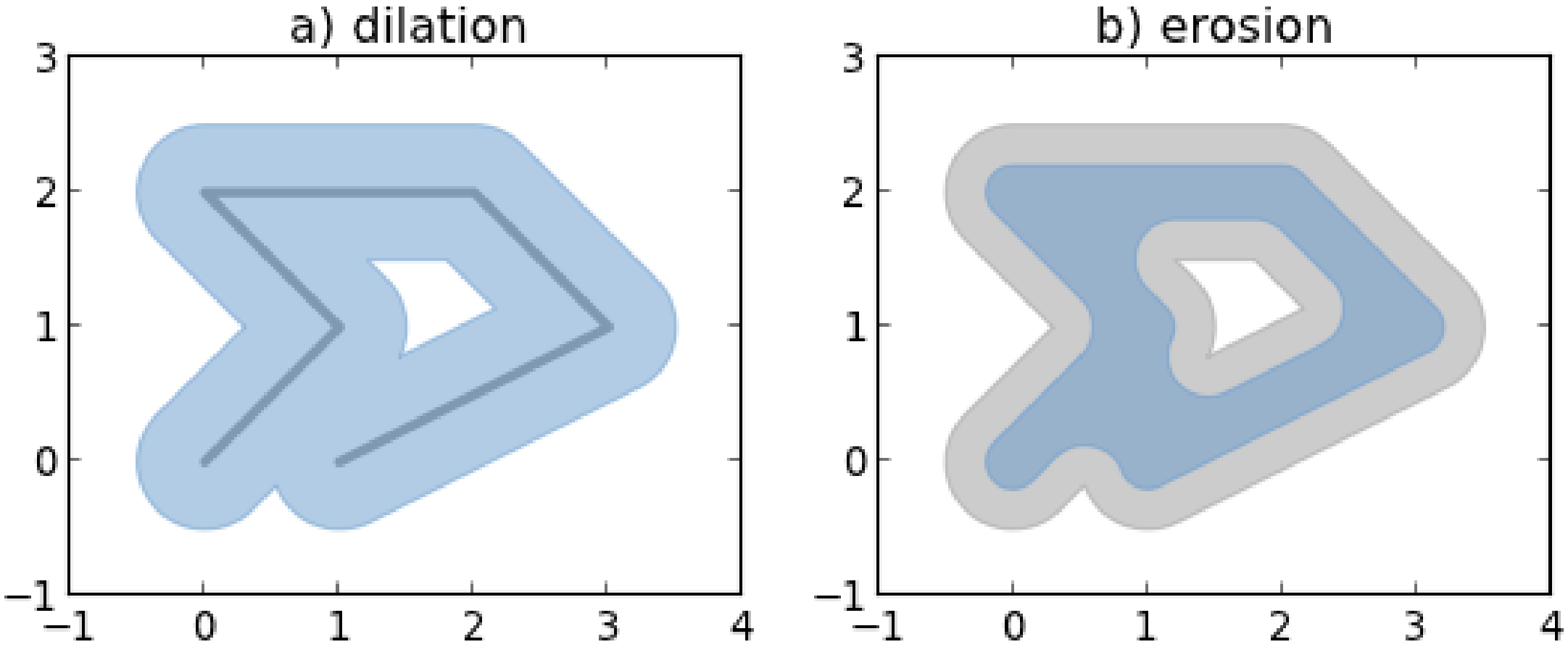
http://kjordahl.github.io/SciPy-Tutorial-2015/#1

Shapely



http://kjordahl.github.io/SciPy-Tutorial-2015/#1

Shapely



http://kjordahl.github.io/SciPy-Tutorial-2015/#1

Shapely Binary predicates

object.almost_equals(other[, decimal=6])

object.contains(other)

object.crosses(other)

object.disjoint(other)

object.equals(other)

object.intersects(other)

object.touches(other)

object.within(other)

http://kjordahl.github.io/SciPy-Tutorial-2015/#1

Spatial Reference Systems

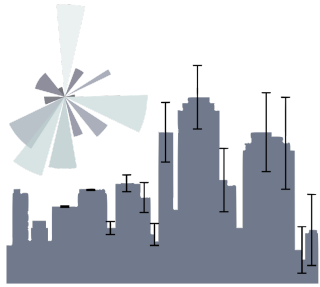
Examples:

- EPSG:4326 latitude, longitude in WGS-84 coordinate system
- EPSG:900913 and EPSG:3857 Google spherical Mercator
- ESRI:102718 NAD 1983 StatePlane New York Long Island FIPS 3104 Feet

Create an SRS with pyproj:

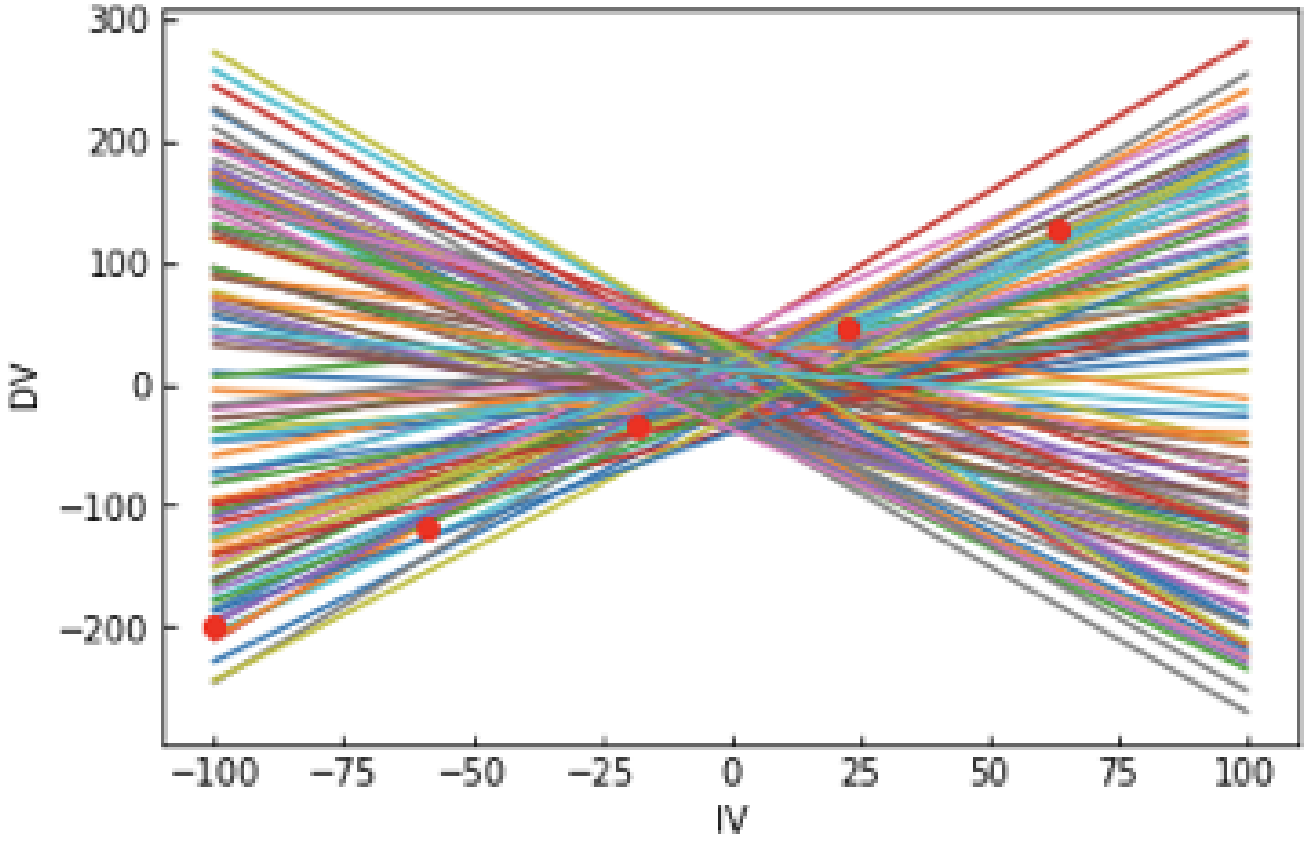
```
1 # convert to another crs
2 world = world.to_crs("EPSG:3395")
```

what if there is no crs to start with?



2 what is a model?

line model: $ax+b$



1

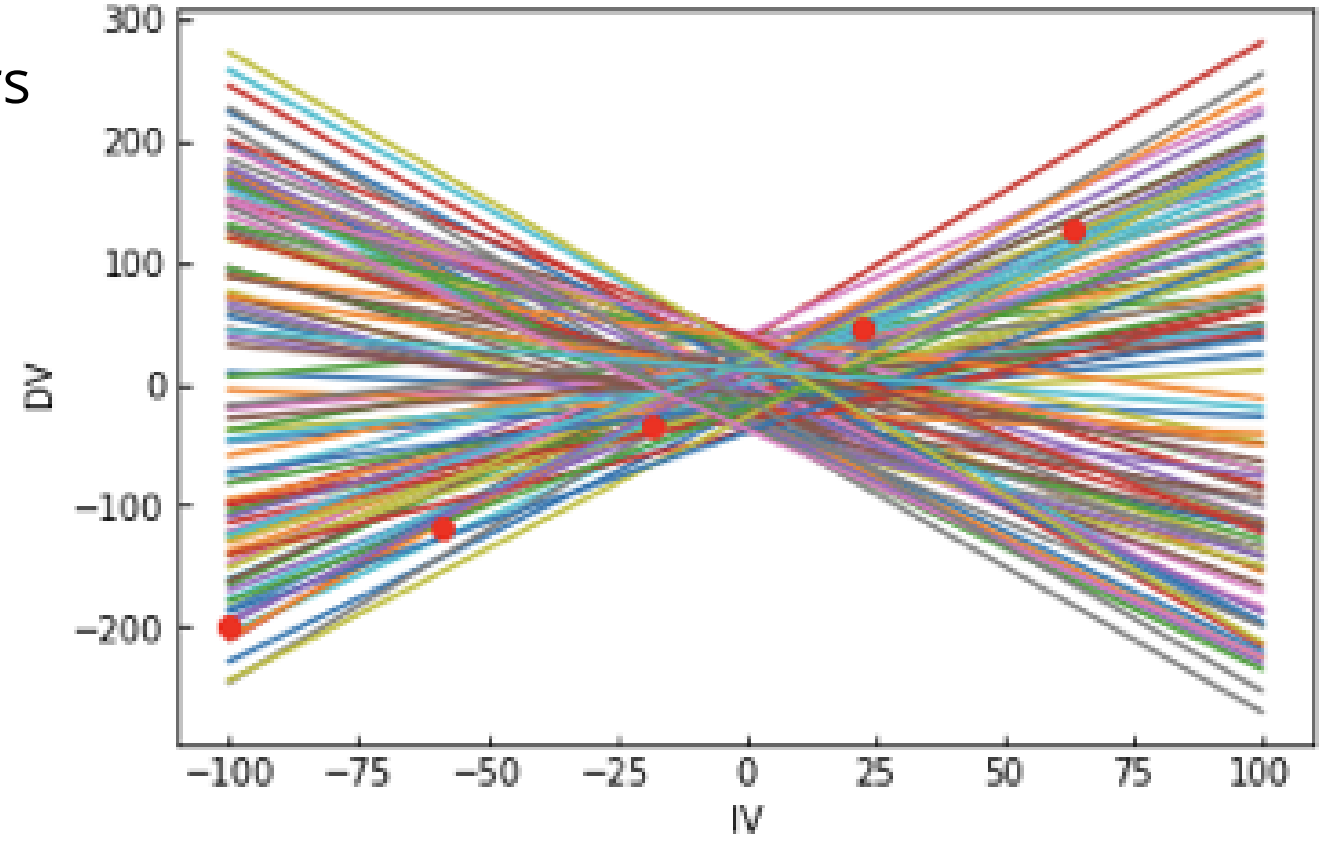
choose your model :

choose a mathematical formula to represent the behavior you see/expect in the data

line model: $ax+b$

a mathematical formula describes a family of shapes. The parameters define the exact shape: in a line fit the parameters are...

- a: slope
- b: intercept



1

choose your model :

choose a mathematical formula to represent the behavior you see/expect in the data

generalizatoin to a polynomial fit:

- the *degree N* of the polynomial is a *hyperparameter* of the model

we choose hyperparameters,

we fit parameters

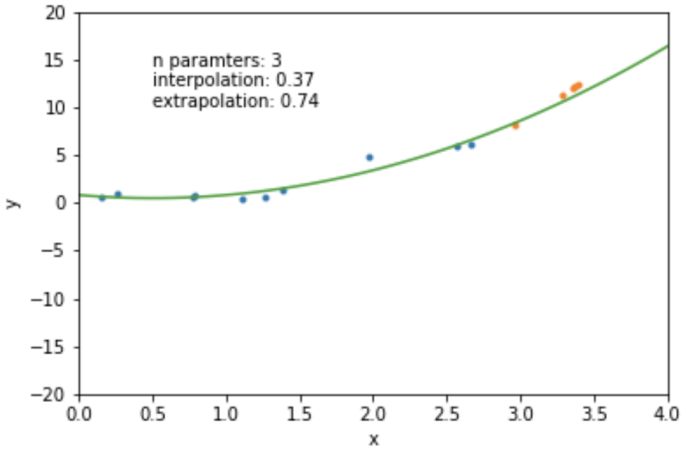
1

choose your model :

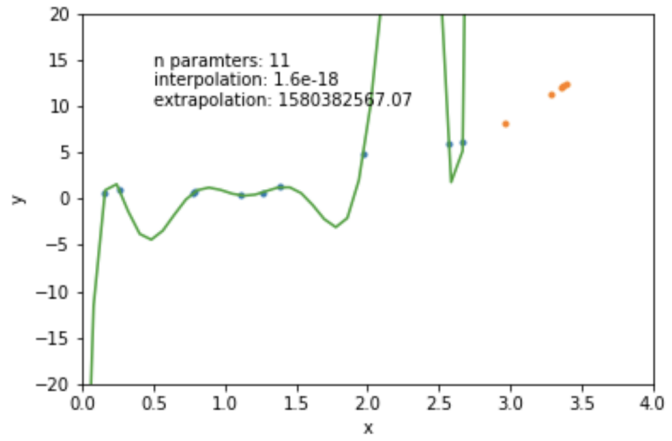
choose a mathematical formula to represent the behavior you see/expect in the data

polynomial model:

$$polyn = \sum_{i=0}^{N} c_i x^i$$



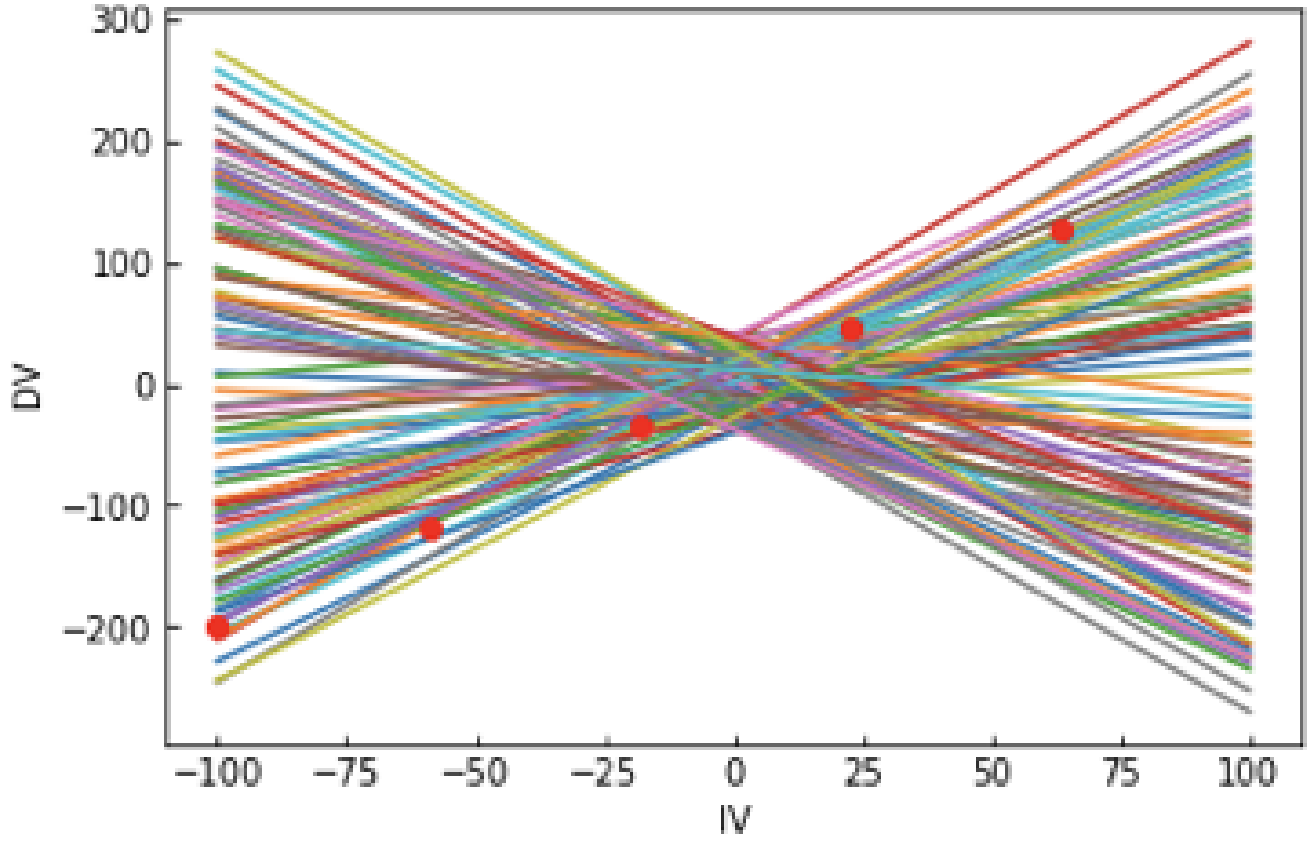
N=2: a conservative hyperparameter choice



this model goes exactly through each data point but it has too many parameters- N = number of data points

https://github.com/fedhere/PUS2020_FBianco/blob/master/classdemo/overfit_animation.ipynb

line model: $ax+b$



2

choose an objective function :

you need a plan to choose the parameters of the model: to "optimize" the model.
You need to choose something to be MINIMIZED or MAXIMIZED

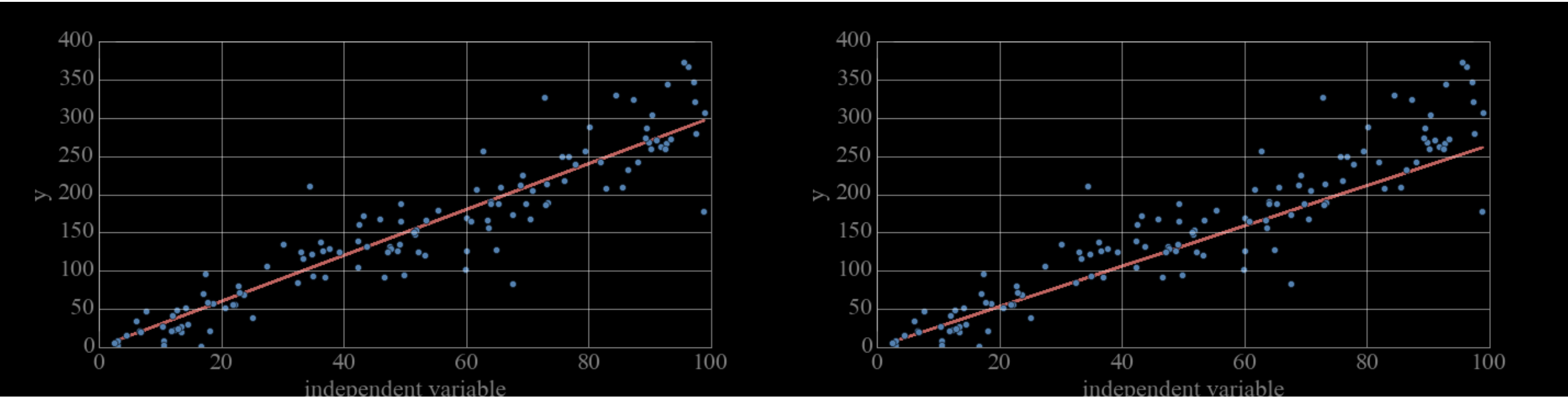
objective function:

what you want to optimize for

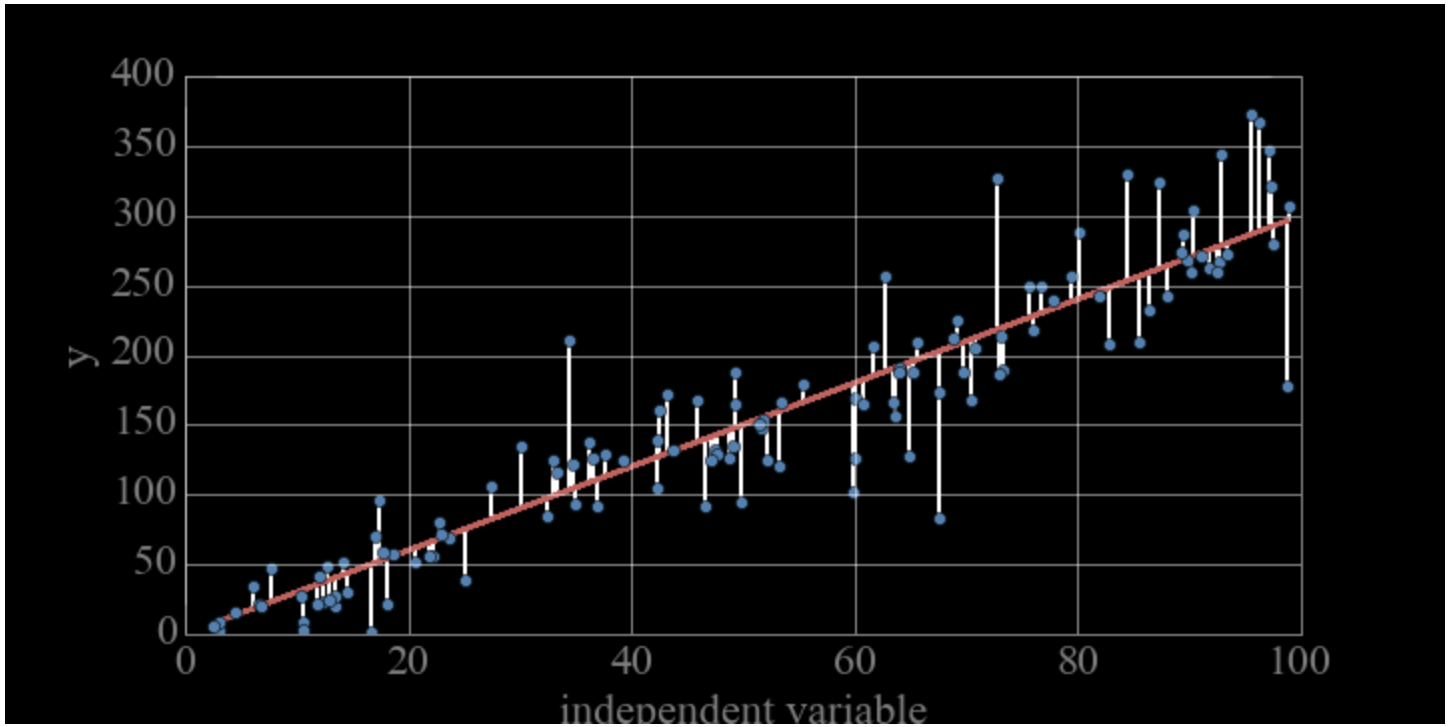
In principle, there are many choices for objective function. But the only procedure that is truly justified—in the sense that it leads to interpretable probabilistic inference, is to make a generative model for the data.

objective function:

what you want to optimize for



objective function:



objective function:

what you want to optimize for

e.g. Sum of residual squared (*least square fit method*)

$$SSE = \sum (y_i - (mx_i + b))^2$$

yi: i-th observation

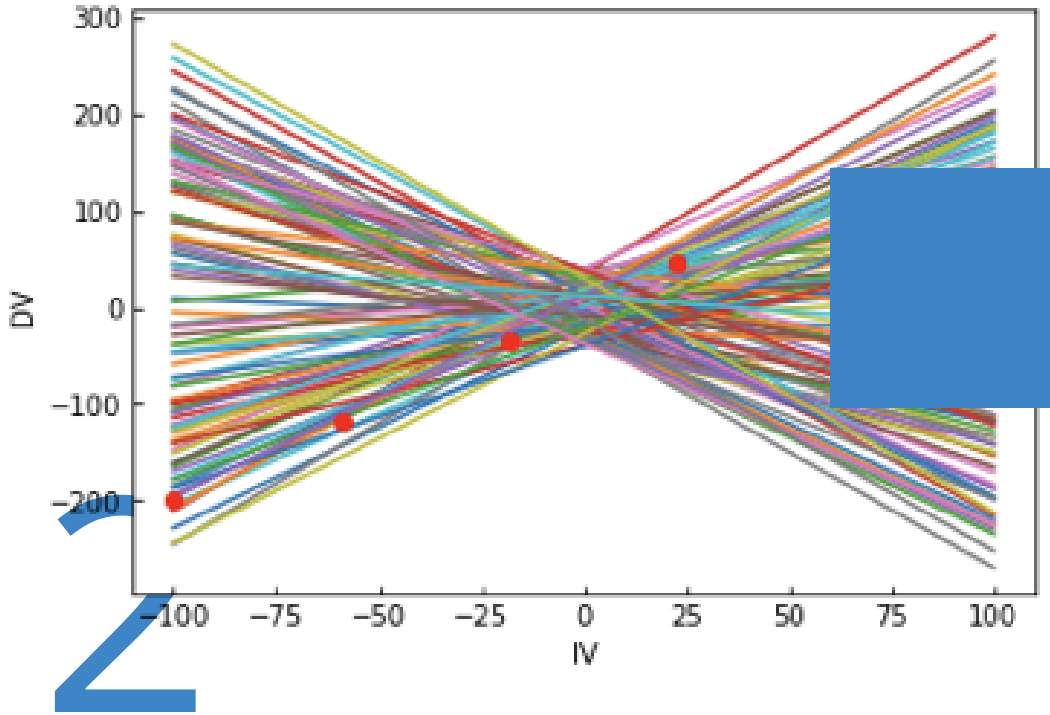
xi: i-th measurement "location"

$$SSE = \sum (y_{i,observed} - y_{i,predicted})^2$$

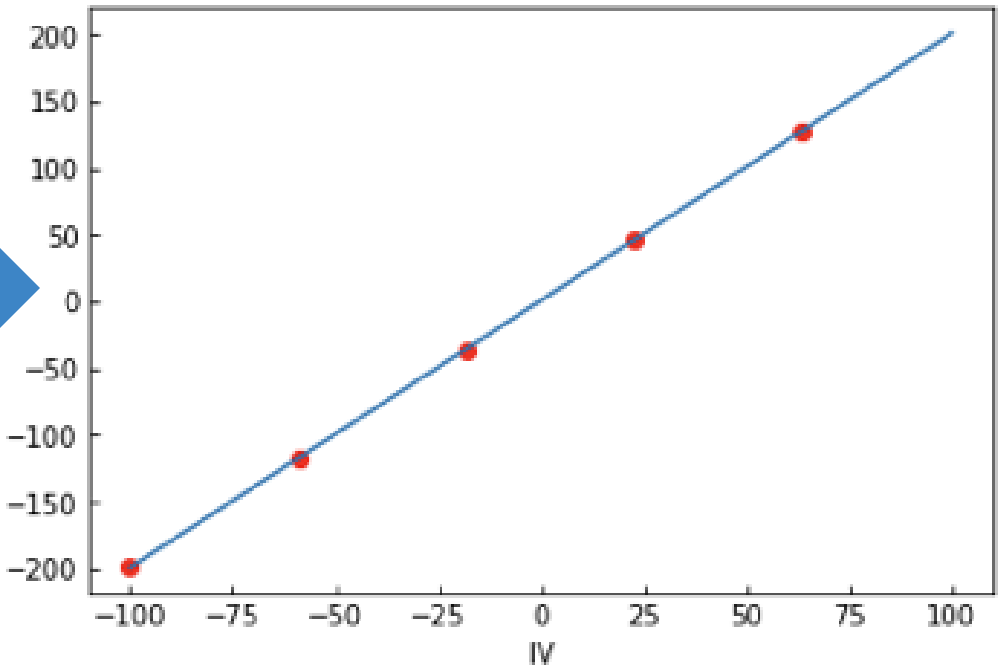
Fit model parameters <==> minimize the Sum of residuals squared

line model: *ax+b*

a line is a family of models

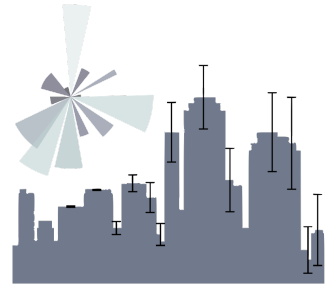


a line with set parameters is a models



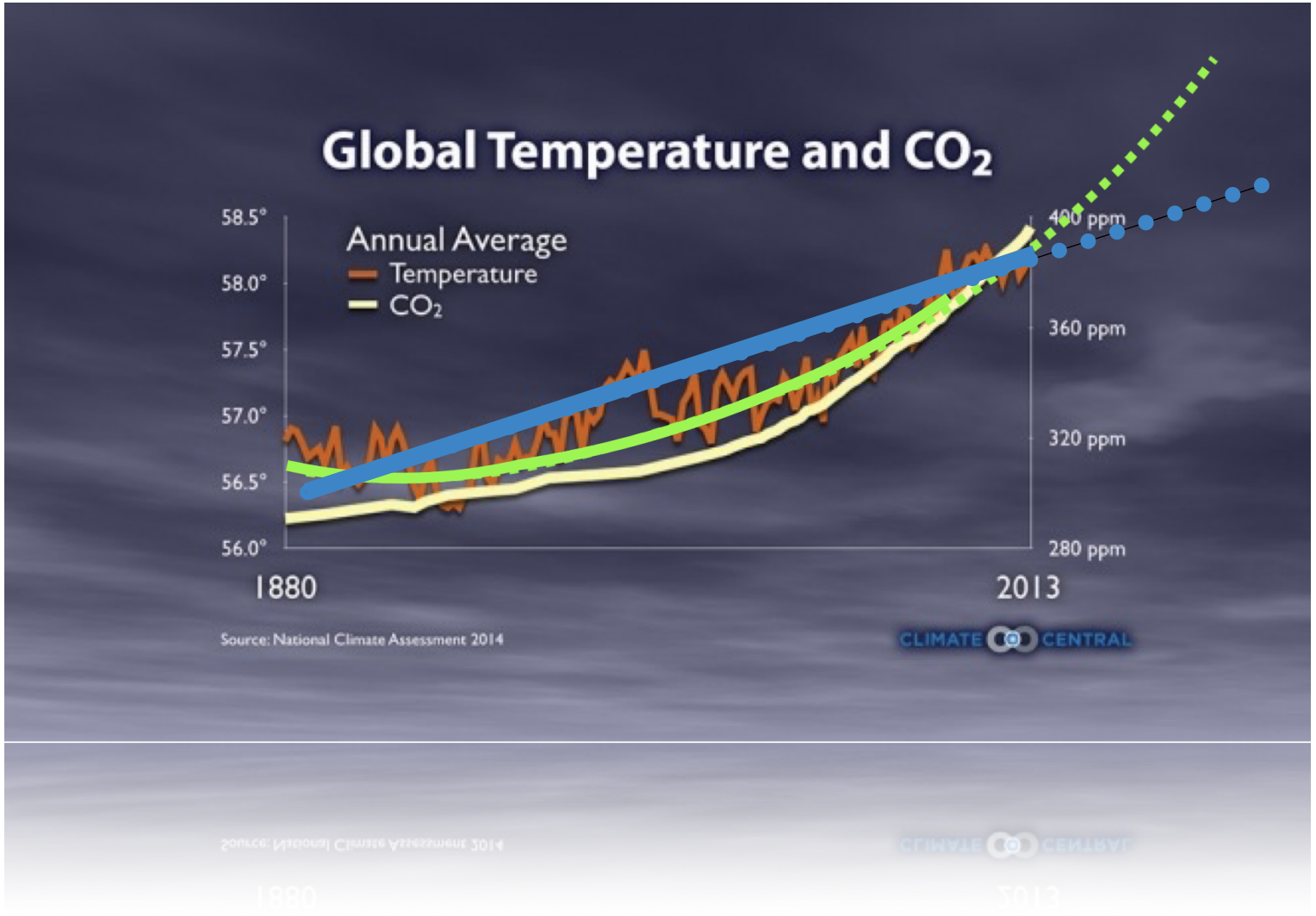
choose an objective function :

you need a plan to choose the parameters of the model: to "optimize" the model.
You need to choose something to be MINIMIZED or MAXIMIZED



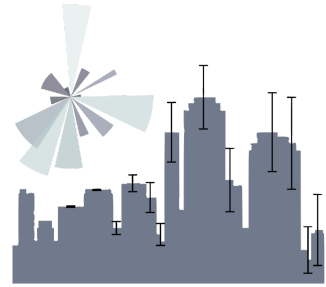
21 why do we model?

why do we model?



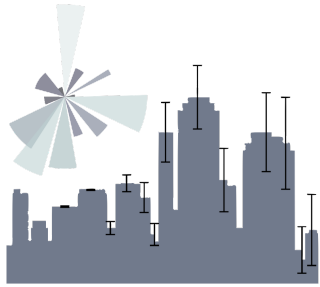
to explain

to predict



3 epistemological rooots of overfitting

Okham's razor

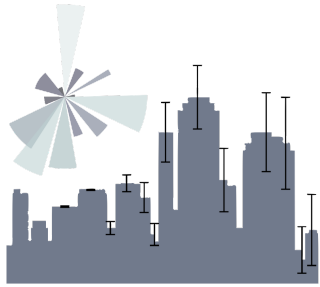


Ockham's razor: ***Pluralitas non est ponenda sine neccesitate***
or "the law of parsimony"

William of Ockham (logician and Franciscan friar) 1300ca
but probably to be attributed to John Duns Scotus

"Complexity needs not to be postulated without a need for it"
"Between 2 theories choose the simpler one"

Okham's razor



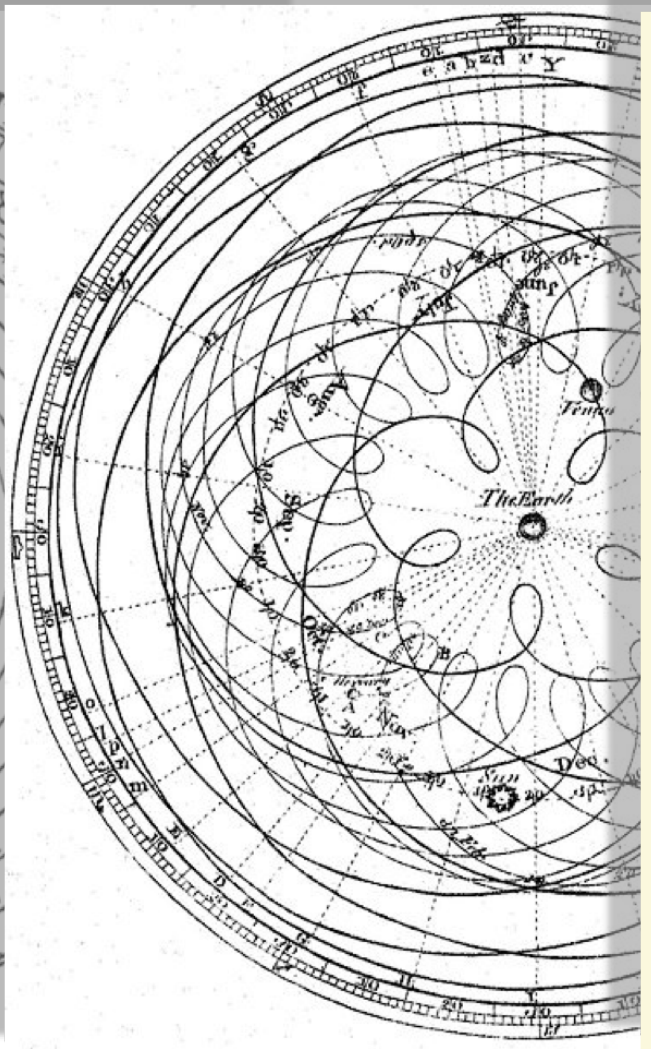
Ockham's razor: ***Pluralitas non est ponenda sine neccesitate***
or "the law of parsimony"

William of Ockham (logician and Franciscan friar) 1300ca
but probably to be attributed to John Duns Scotus

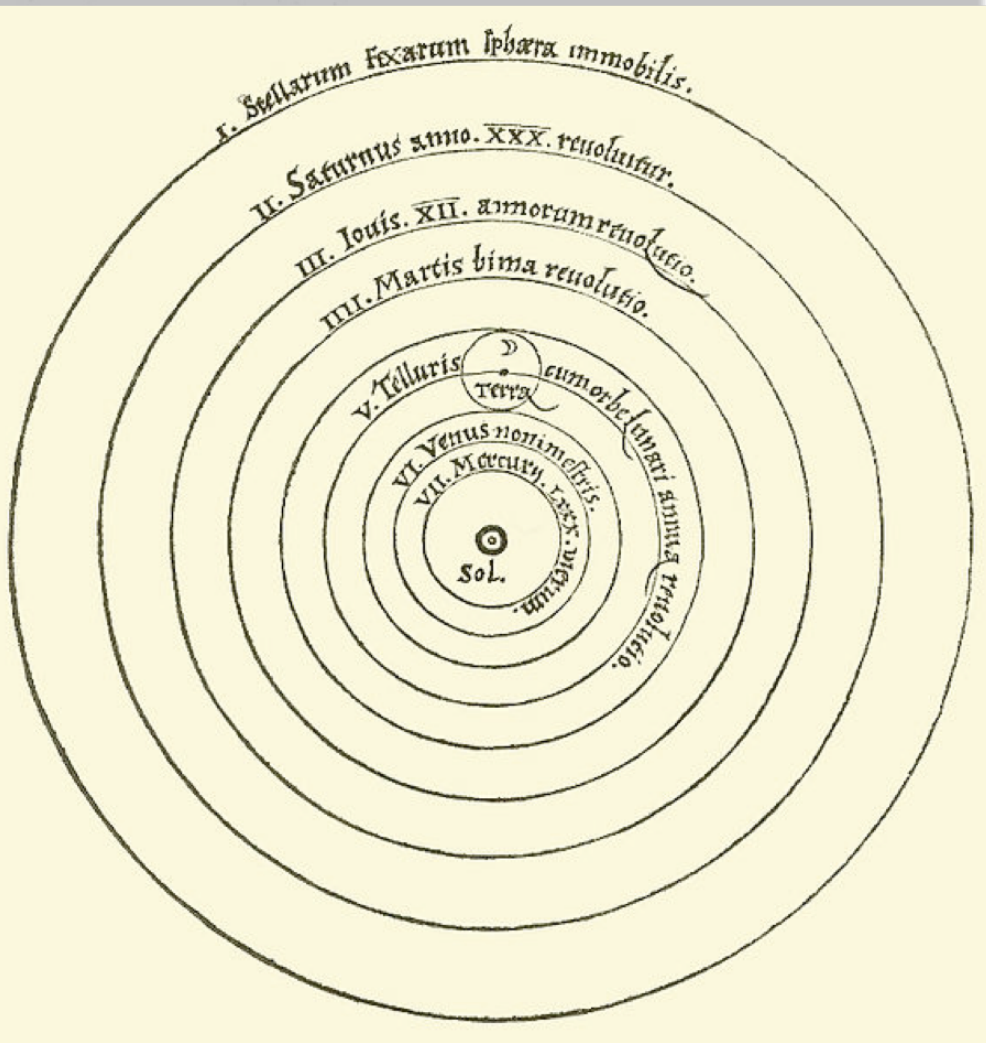
"Complexity needs not to be postulated without a need for it"
"Between 2 theories choose the simpler one"
"Between 2 theories choose the one with fewer parameters"

Okham's razor

Peter Apian, Cosmographia, Antwerp, 1524



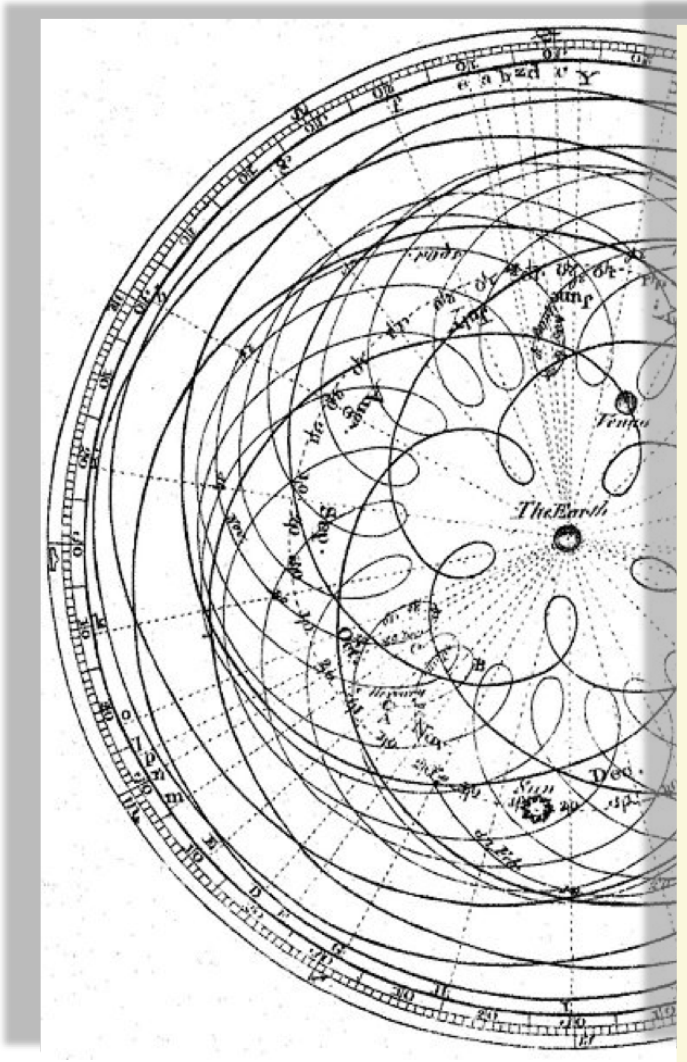
Author Dr Long's copy of Cassini, 1777



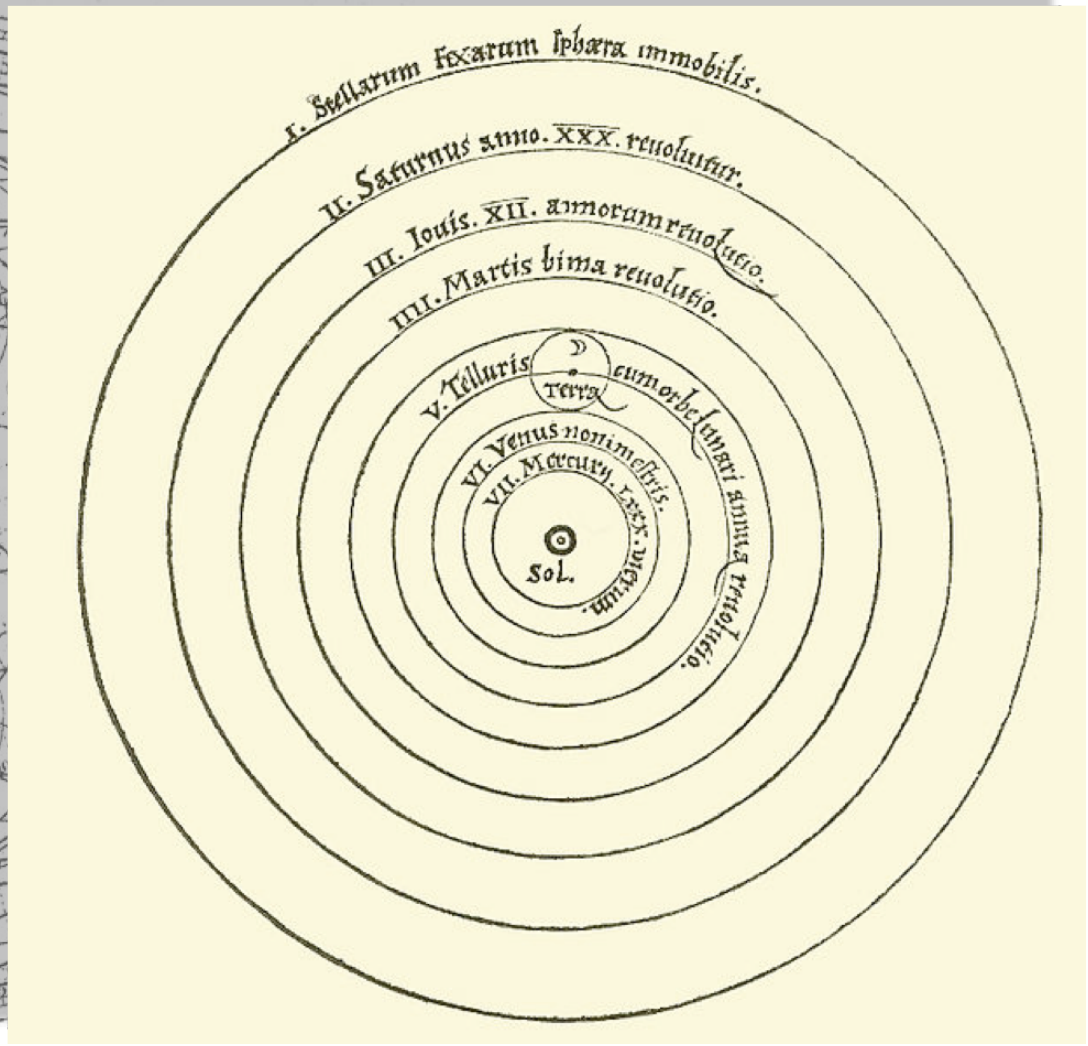
Heliocentric model from Nicolaus Copernicus' "De revolutionibus orbium coelestium".

Okham's razor

Two theories may explain a phenomenon just as well as each other. In that case you should prefer the simpler one

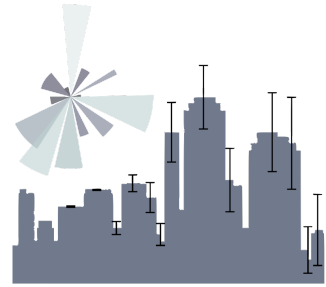


Author Dr Long's copy of Cassini, 1777

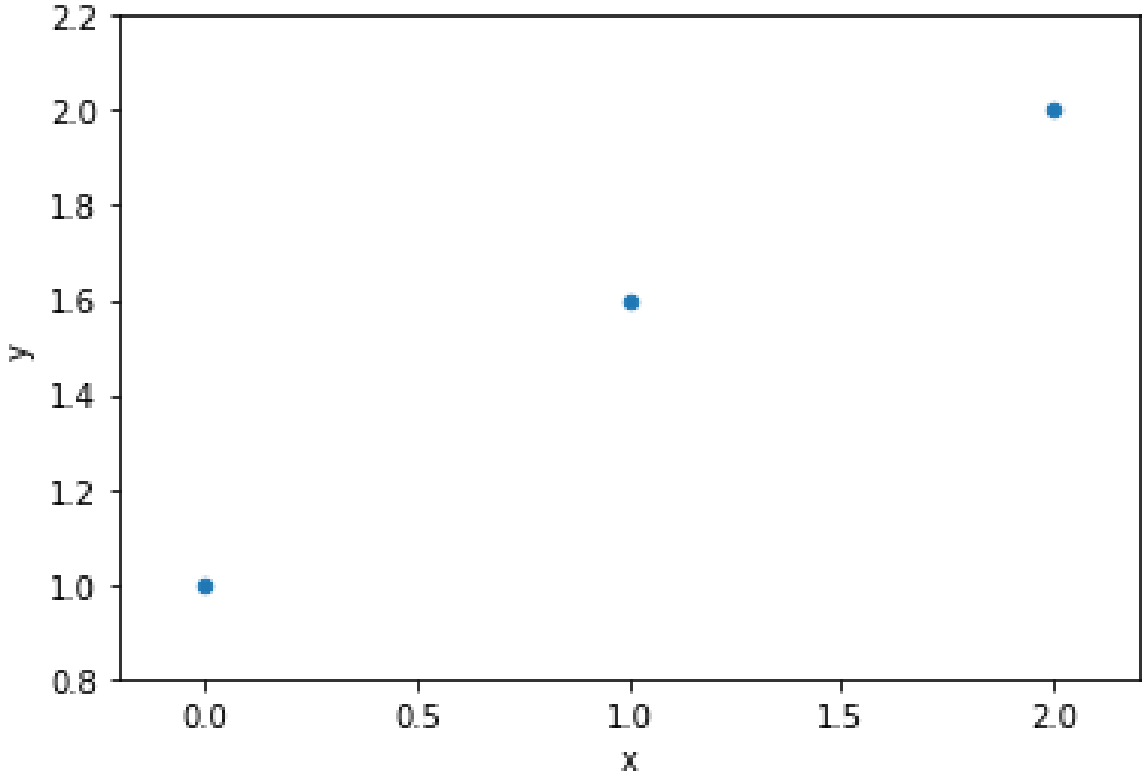


Heliocentric model from Nicolaus Copernicus' "De revolutionibus orbium coelestium".

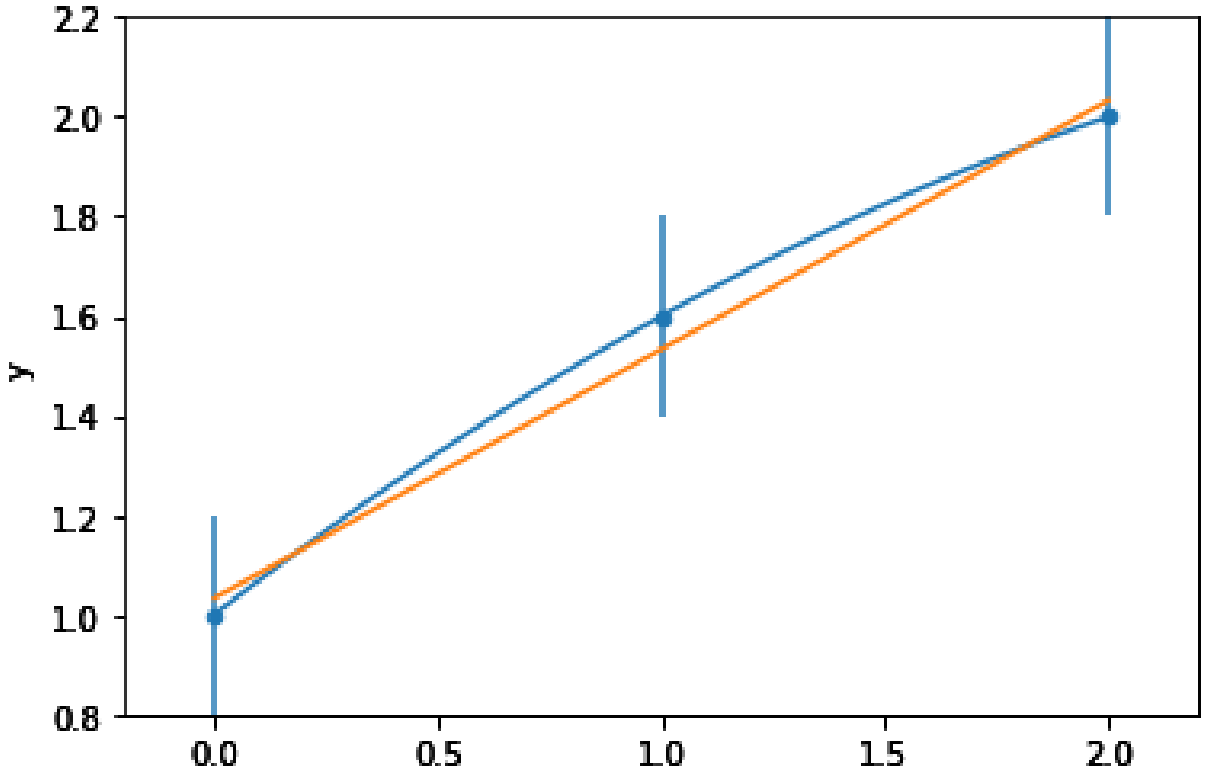
Okham's razor



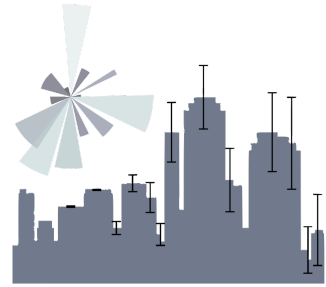
data



model fit to data



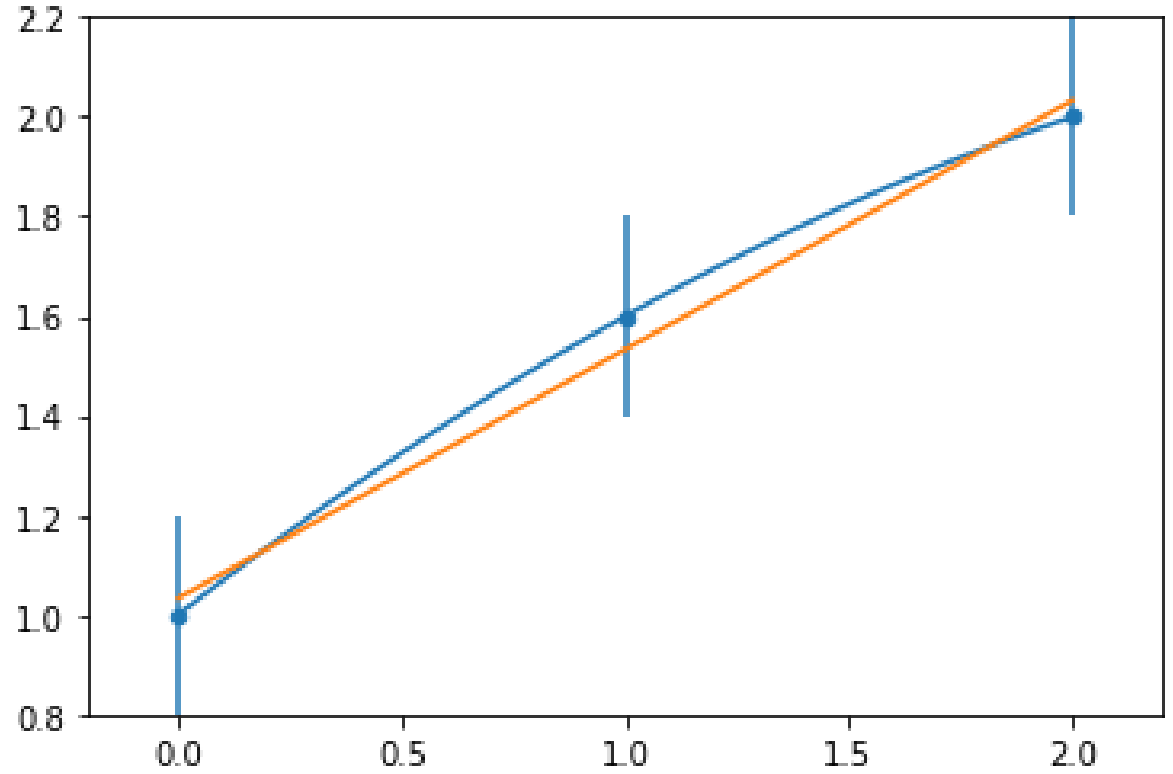
Okham's razor



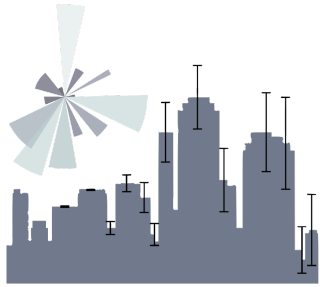
model fit to data

$y = ax + b$

$y = ax^2 + bx + c$



Okham's razor

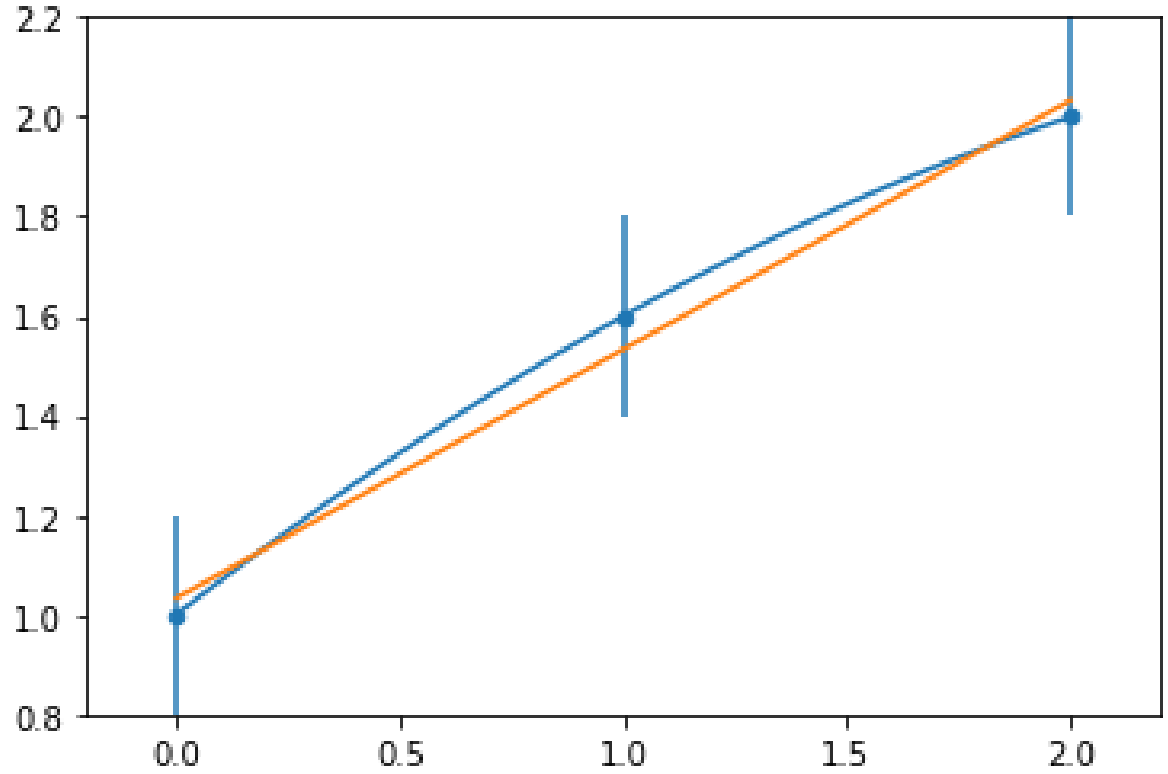


model fit to data

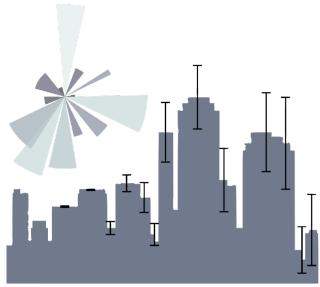
$y = ax + b$

$y = ax^2 + bx + c$

1 variable: x



Okham's razor



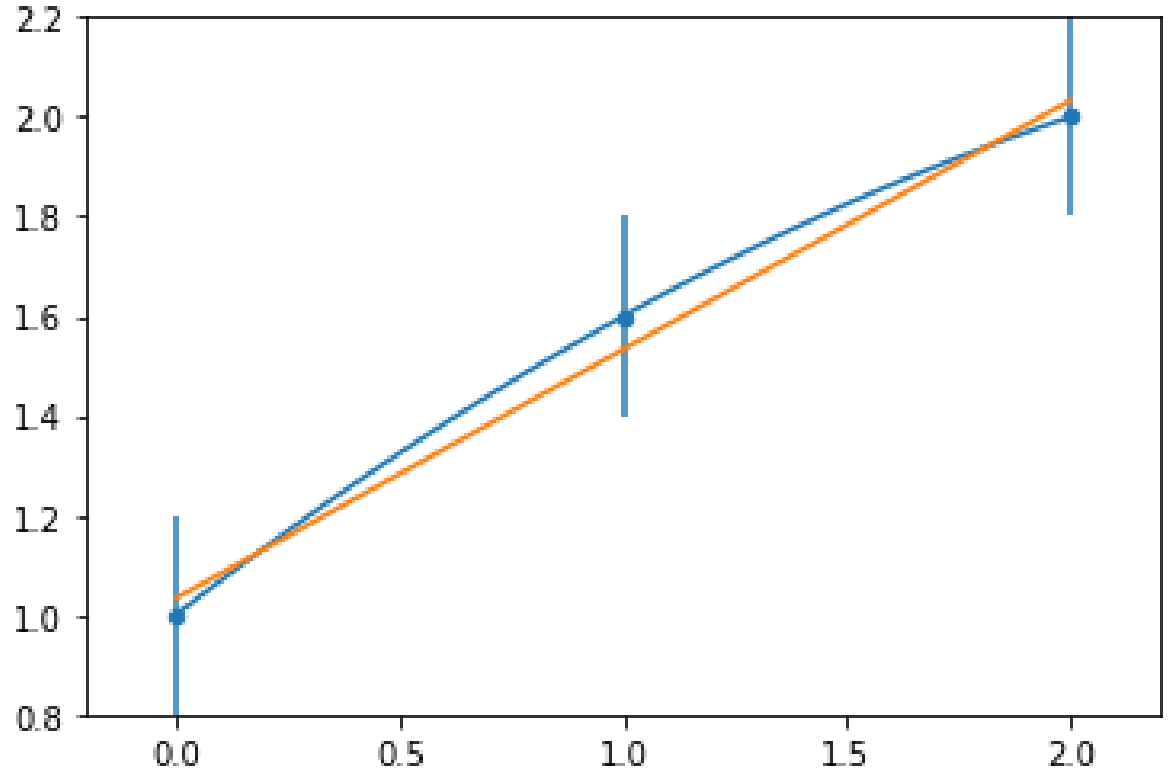
the complexity of a model can be measured by the number of variables and the numbers of parameters

model fit to data

$y = 0x + 0$

$y = 0x^2 + 0x + 0$

parameters



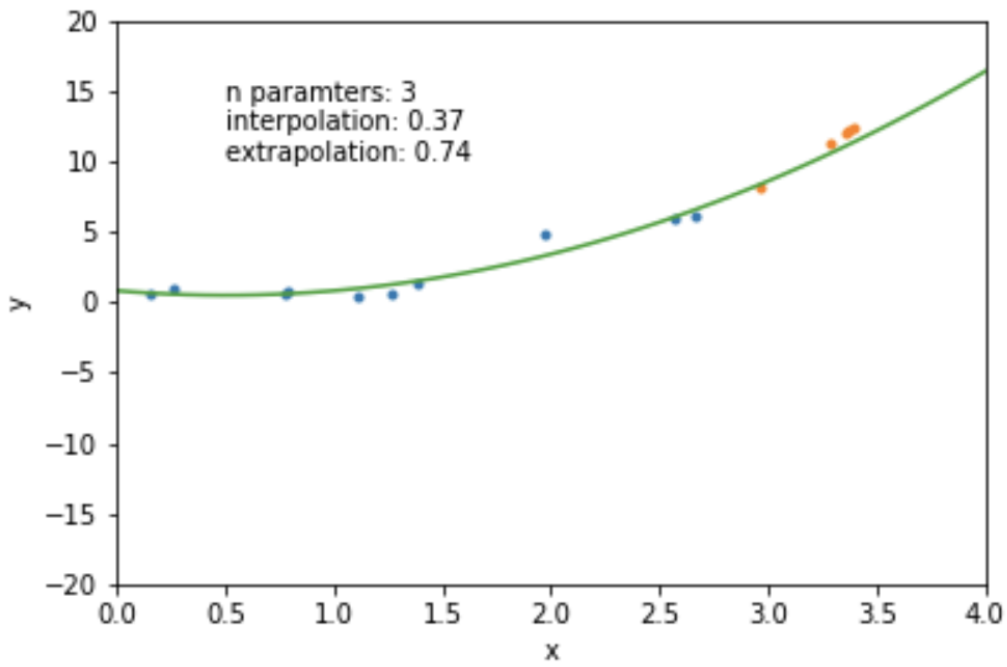
Okham's razor

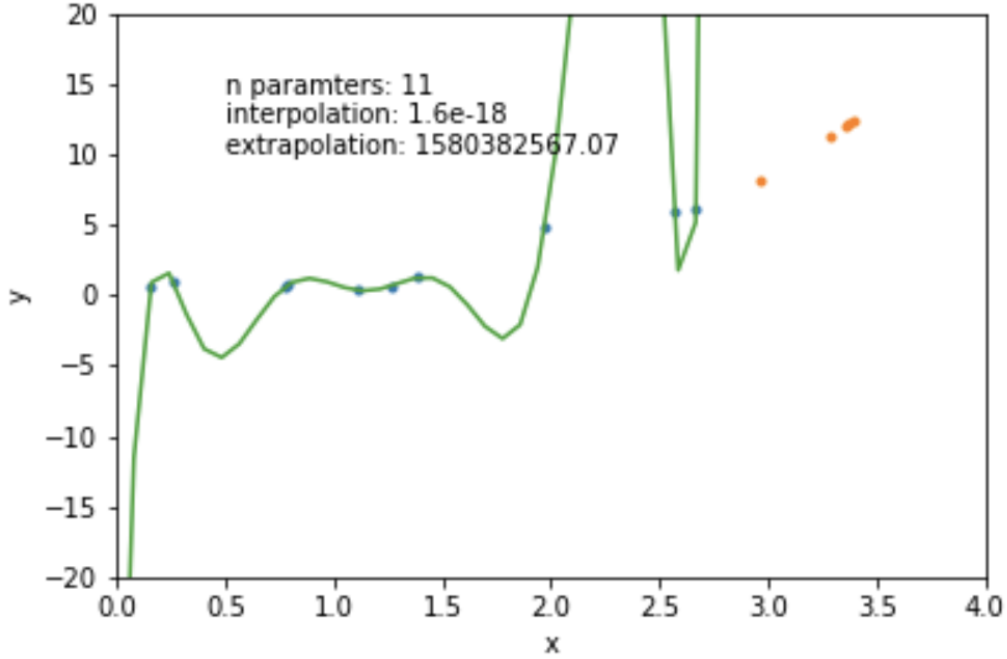
the complexity of a model can be measured by the number of variables and the numbers of parameters

mathematically: given N data points there exist an N-features model that goes exactly through each data point. but is it useul??

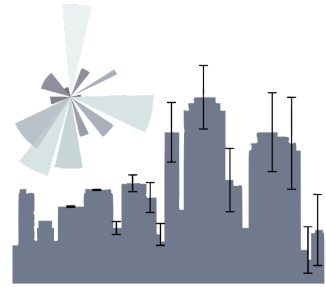
Overfitting: fitting data with a model that is too complex and that does not extend to new data (low predictive power on test data)

https://github.com/fedhere/PUS2020_FBianco/blob/master/classdemo/overfit_animation.ipynb

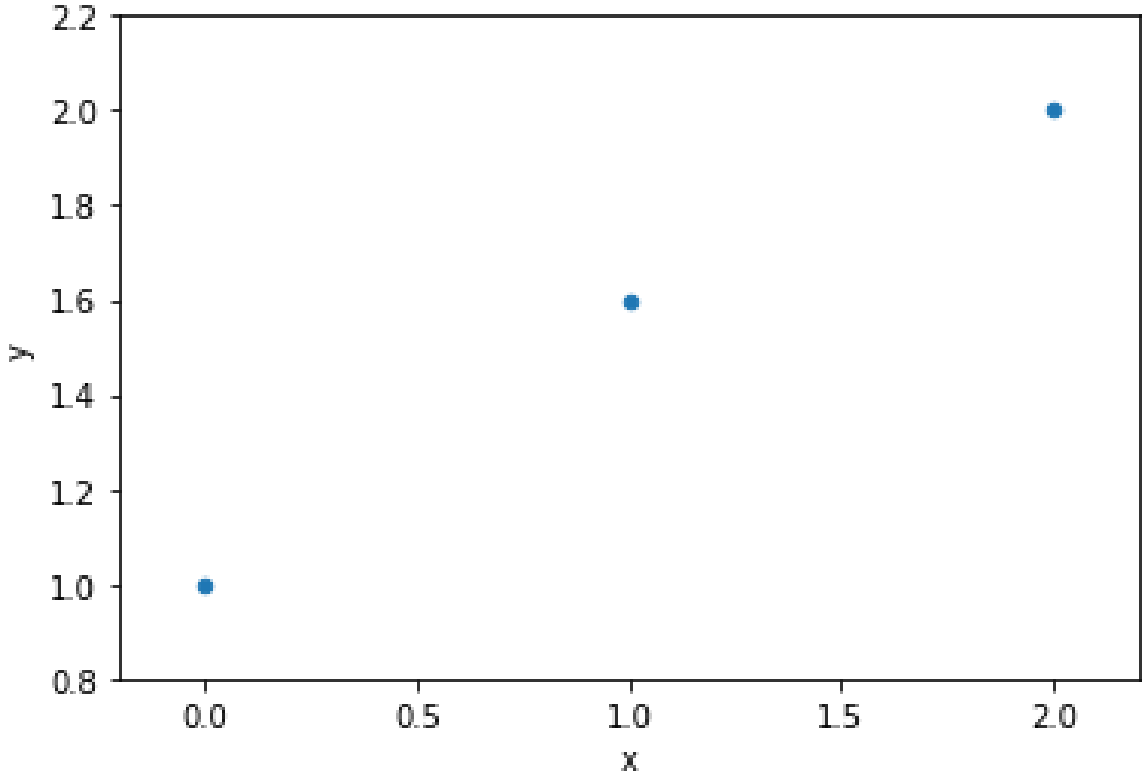




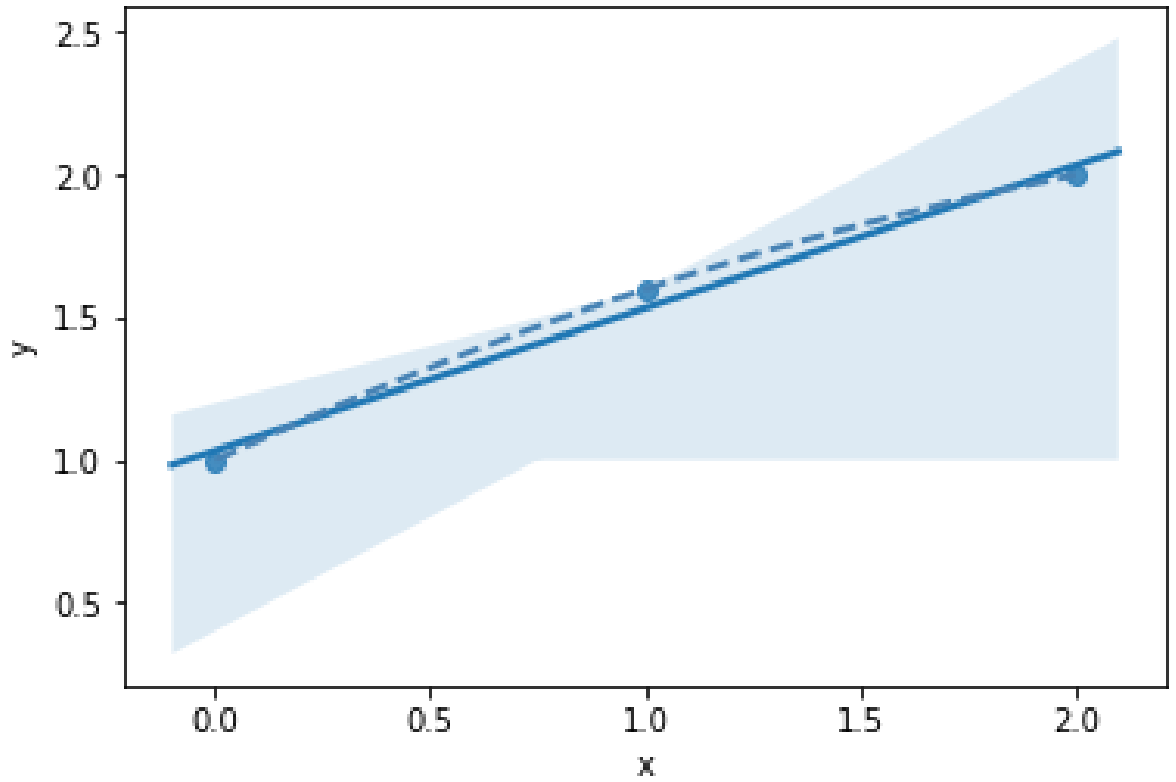
Okham's razor



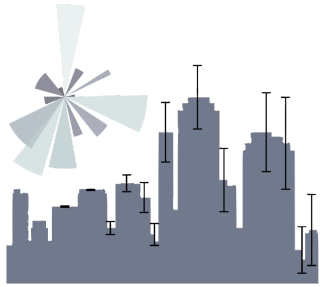
data



model fit to data



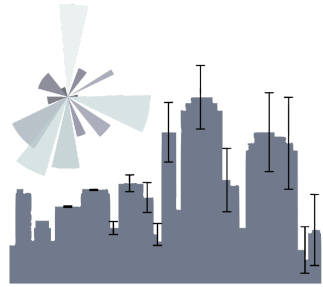
Okham's razor



https://nbviewer.jupyter.org/gist/fedhere/ef2da384b9e114267e8e93b7366e4ff6

Dep. Variable:	y	**R-squared:**	0.987
Model:	OLS	**Adj. R-squared:**	0.974
Method:	Least Squares	**F-statistic:**	75.00
Date:	Mon, 07 Sep 2020	**Prob (F-statistic):**	0.0732
Time:	09:49:48	**Log-Likelihood:**	4.9071
No. Observations:	3	**AIC:**	-5.814
Df Residuals:	1	**BIC:**	-7.617
Df Model:	1		

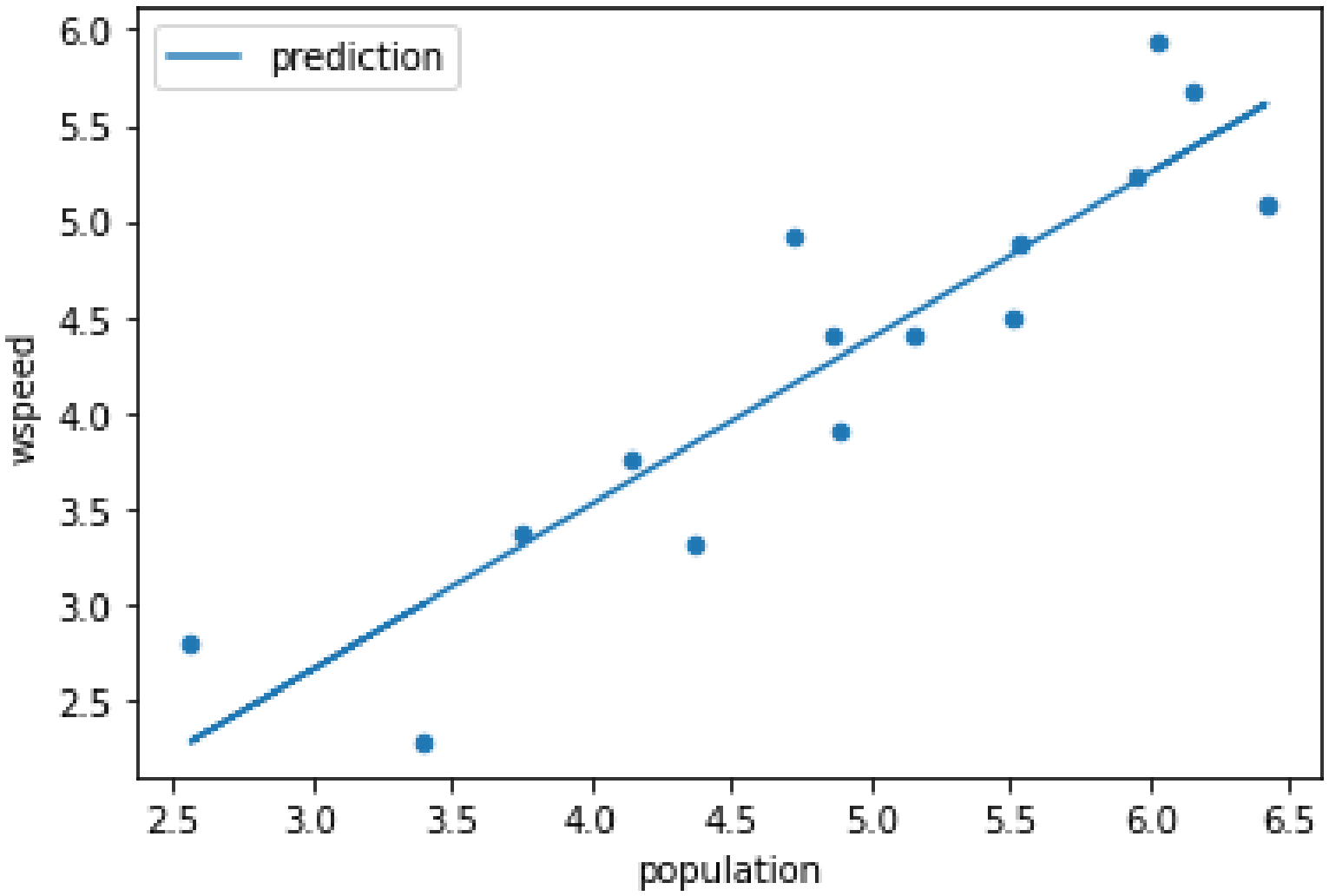
https://nbviewer.jupyter.org/gist/fedhere/ef2da384b9e114267e8e93b7366e4ff6



4 fitting a line to data (the longish story)

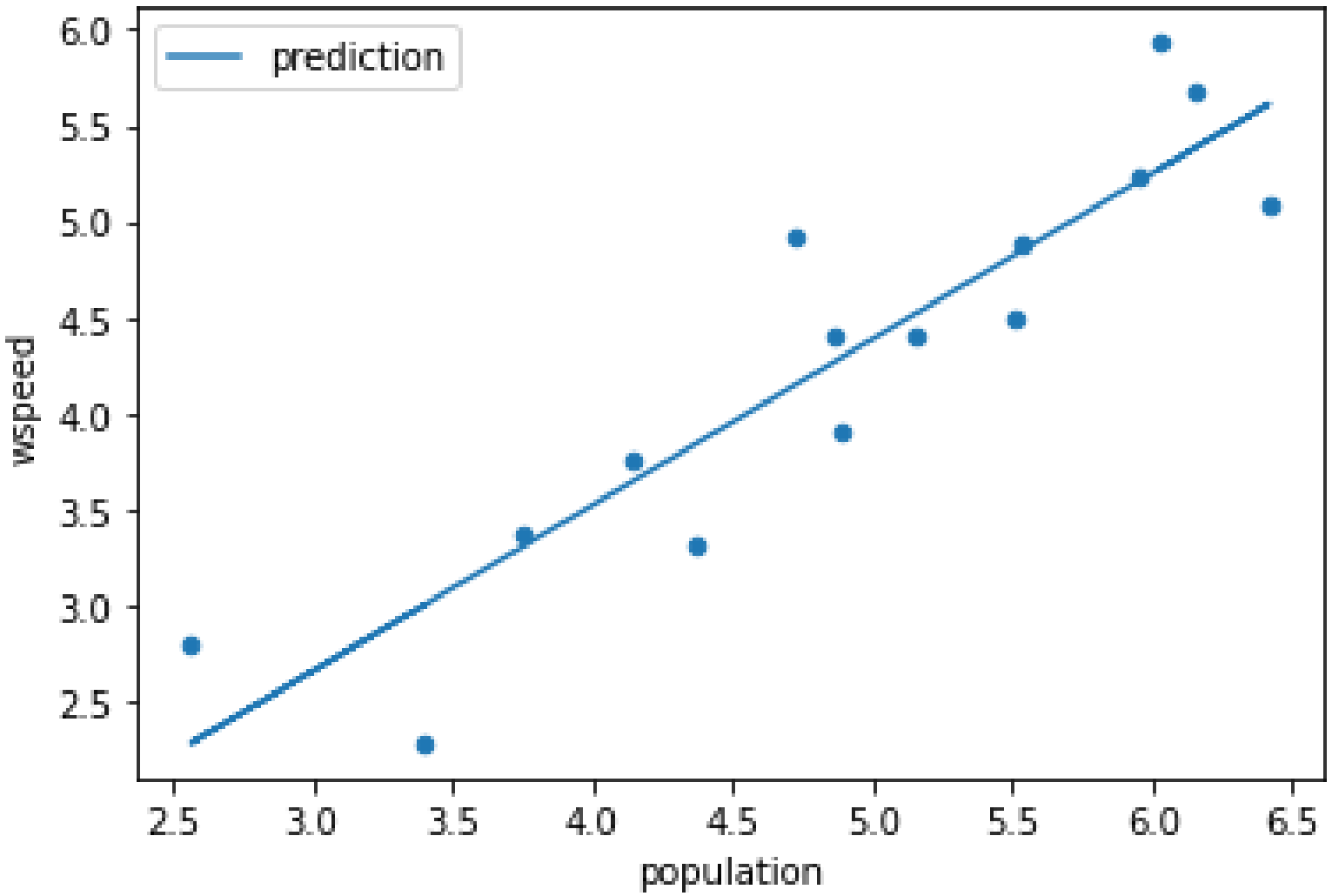
notebook -
https://github.com/fedhere/PUS2020_FBianco/blob/master/classdemo/linear_regression_in_detail.ipynb

data
https://raw.githubusercontent.com/fedhere/PUS2020_FBianco/master/data/walkingsped_Bettencourt07.csv",

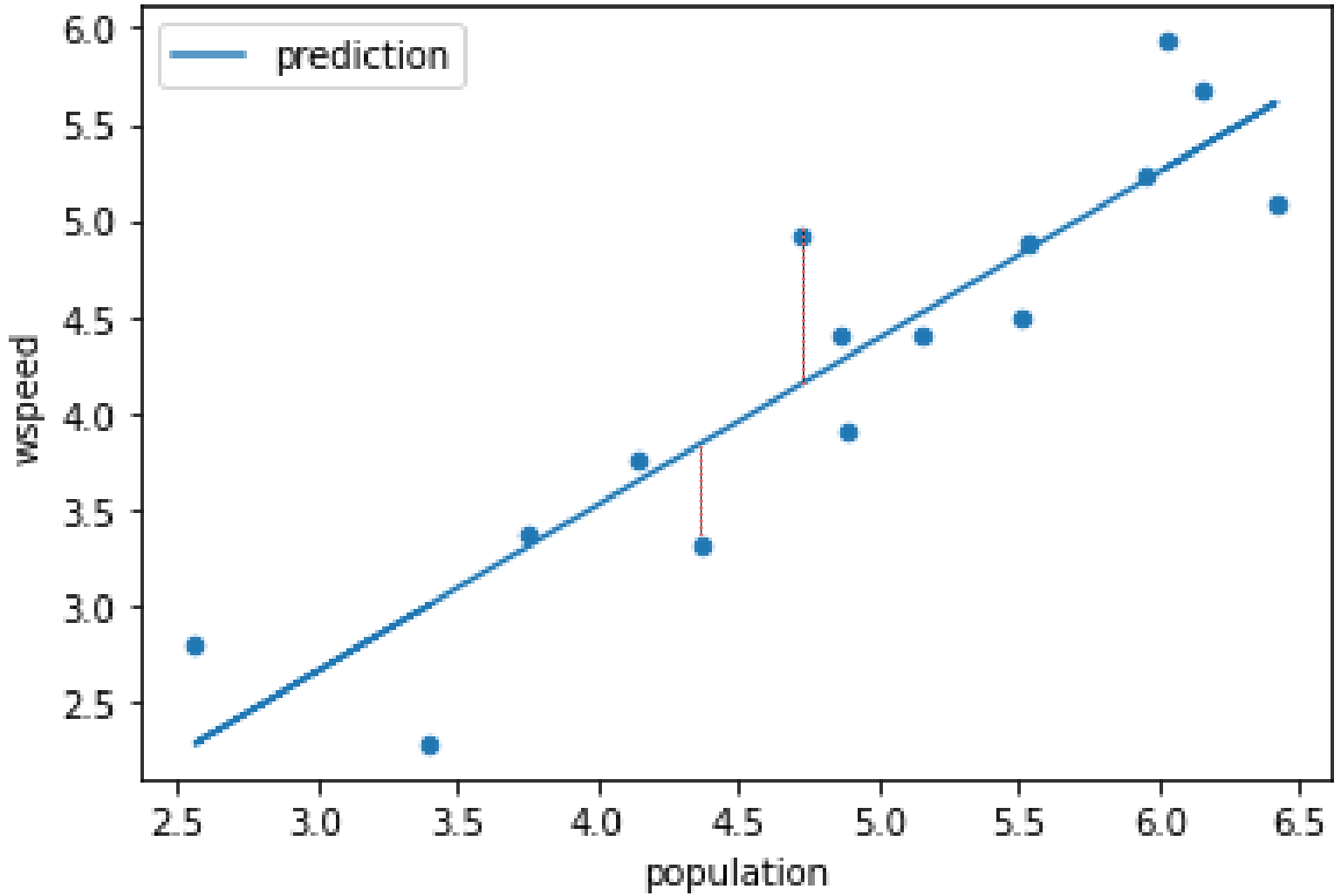


We are trying to find the "best" line that goes through the data... but "best" is a judgment call

Choosing the objective function



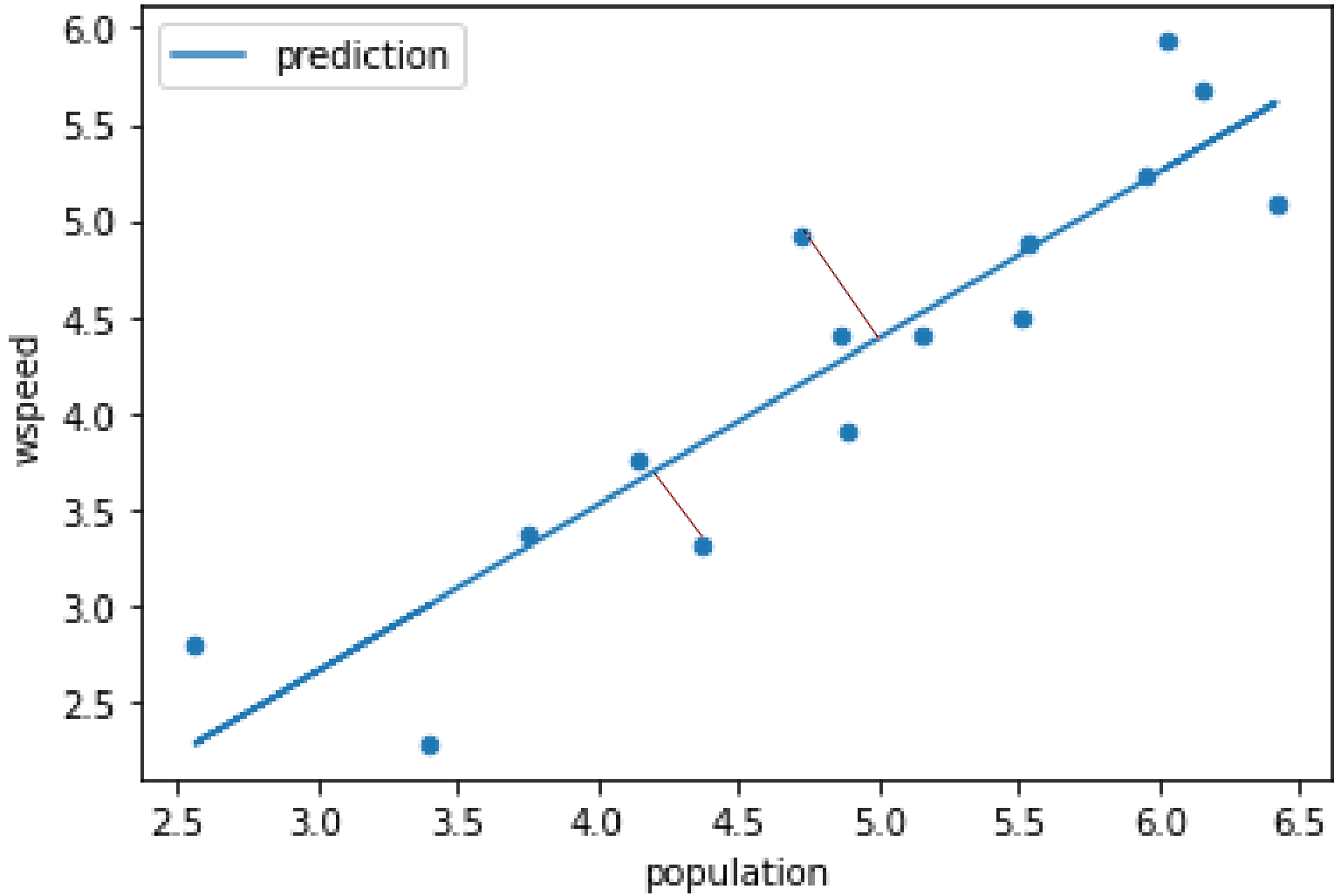
We are trying to find the "best" line that goes through the data... but "best" is a judgment call



We are trying to find the "best" line that goes through the data... but "best" is a judgment call

minimize *something*:

$$\text{Sum of errors} : \sum_{i=0}^{N} y_i - y_{i,predicted}$$



We are trying to find the "best" line that goes through the data... but "best" is a judgment call

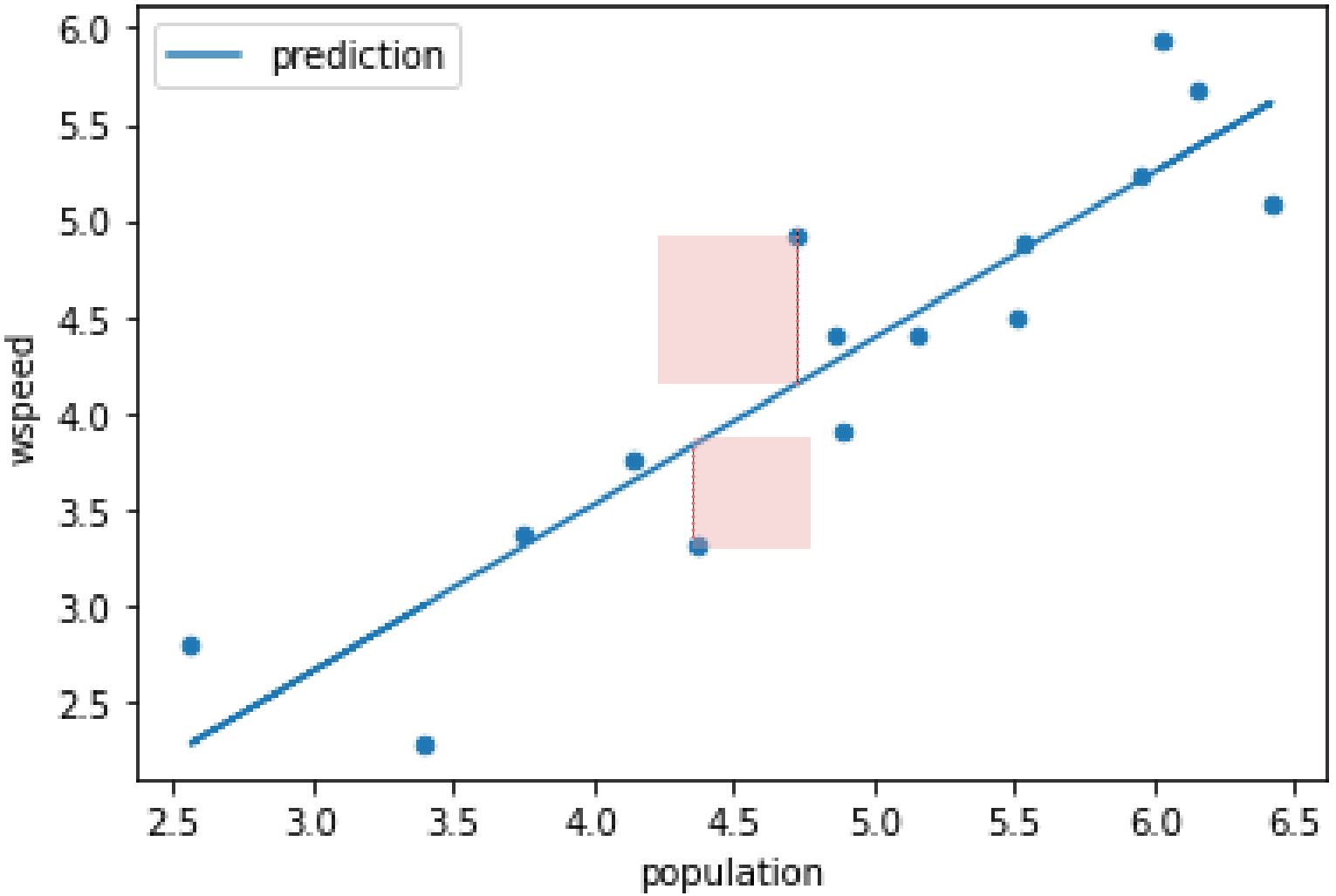
minimize *something*:

$$\text{Sum of errors} : \sum_{i=0}^{N} y_i - y_{i,predicted}$$

why square?

- So that the errors do not cancel themselves out

- To add more weight to predictions that are worse



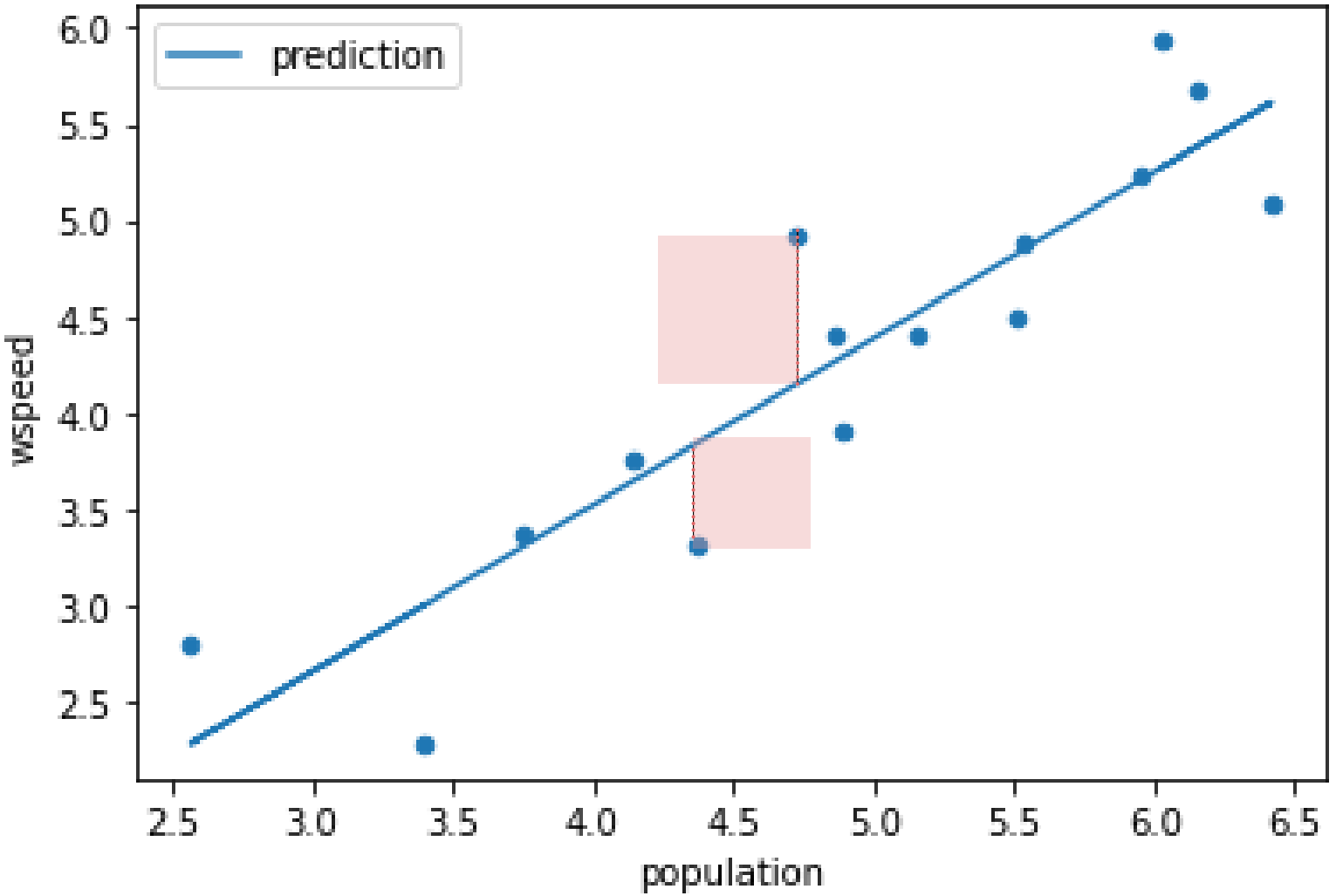
Ordinary least square

minimize *something*:

$$\text{Sum of quared errors} : SSE = \sum_{i=0}^{N} (y_i - y_{i,predicted})^2$$

we can minimize manually...

```
def sumsqerror(y, yp):
  ''' objective function squared error
  y: vector of observations
  yp: vector of predictions
  return: sum squared difference
  '''
  return ((y - yp) ** 2).sum()

minnow = 1e7
for s in np.arange(0, 3, 0.01):
  for i in np.arange(0, 2.5, 0.01):
    prediction = df['population'] * s + i
    sse = sumsqerror(df.wspeed, prediction)
    if sse < minnow:
      minnow = sse
```



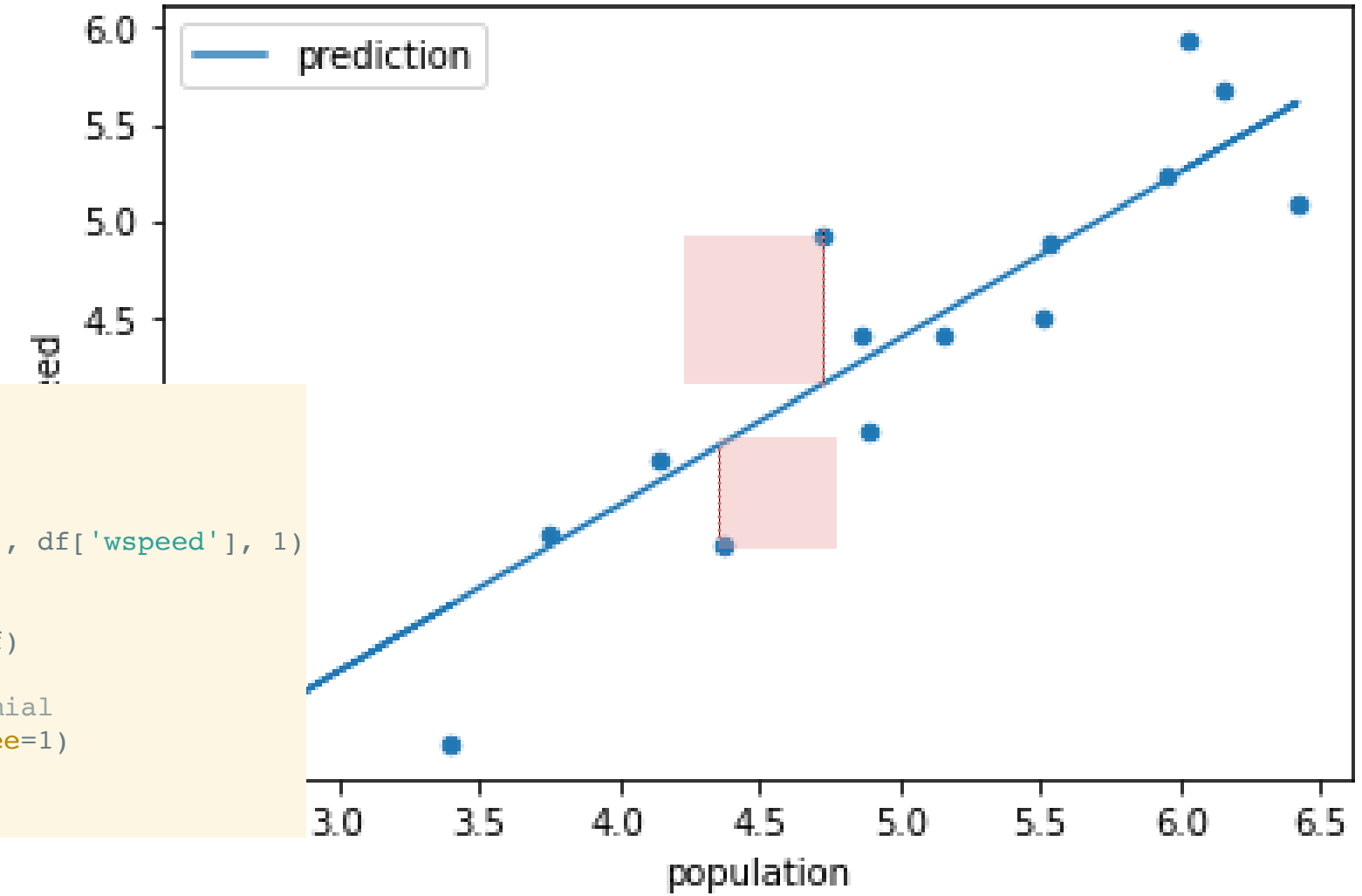
Ordinary least square

minimize *something*:

$$\text{Sum of quared errors} : SSE = \sum_{i=0}^{N} (y_i - y_{i,predicted})^2$$

but its a lot easier to use built in functions

```
# seaborn (just plots)
sns.regplot(df['population'], df['wspeed'])

# numpy
slope, intercept = np.polyfit(df['population'], df['wspeed'], 1)

# statsmodels formula
smf.ols(formula='wspeed ~ population', data=df)

# statsmodels OLS works for any degree polynomial
polynomial_features = PolynomialFeatures(degree=1)
xp1 = polynomial_features.fit_transform(x)
model = sm.OLS(y[:10], xp1[:10]).fit()
```



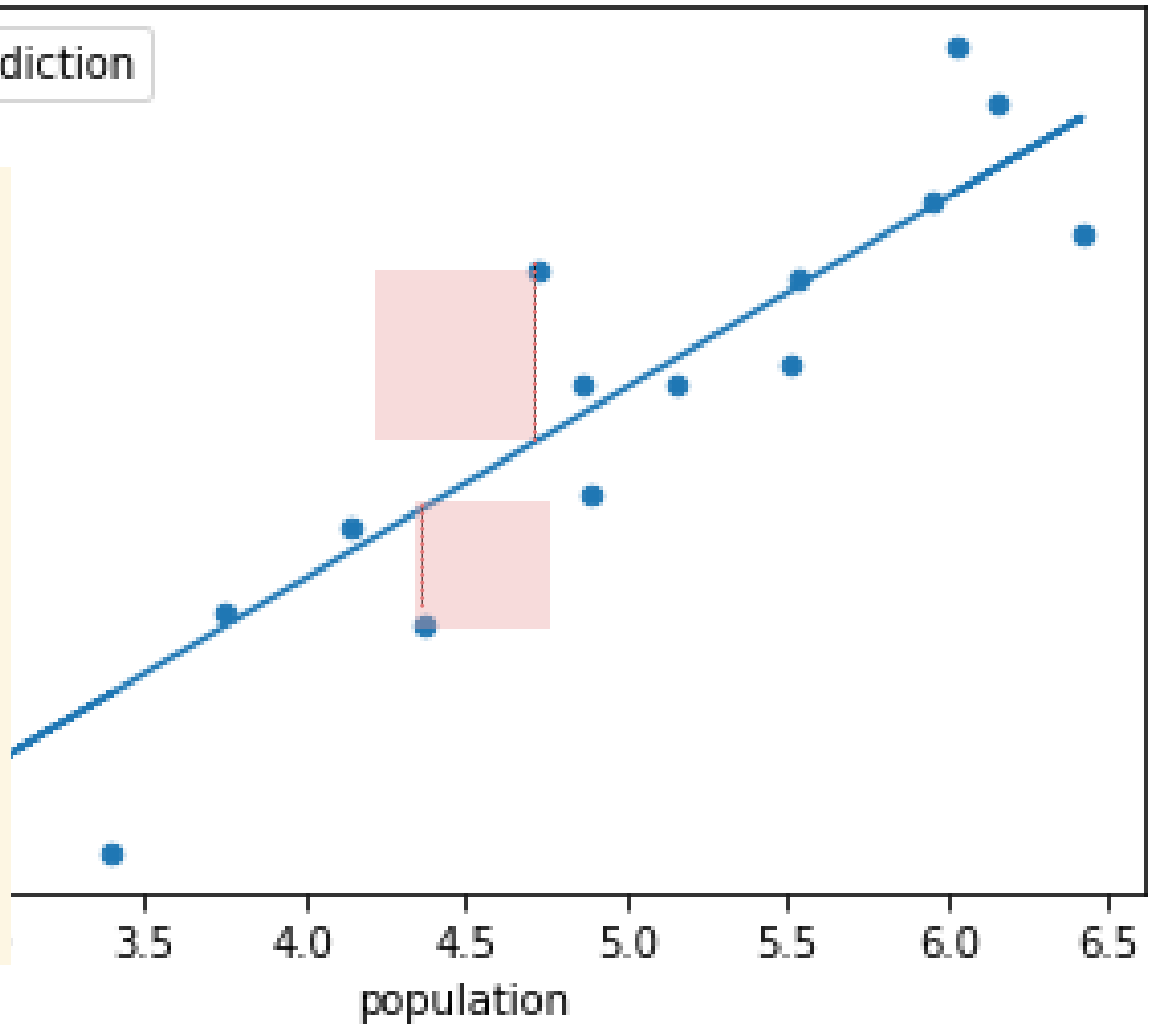
We are trying to find the "best" line that goes through the data... but "best" is a judgment call

minimize *something*:

$$\text{Sum of quared errors} : SSE = \sum_{i=0}^{N} (y_i - y_{i,predicted})^2$$

but its a lot easier to use built in functions

```python
# seaborn (just plots)
import seaborn as sns
sns.regplot(df['population'], df['wspeed'])

# numpy
import numpy as np
slope, intercept = np.polyfit(df['population'], df['wspeed'], 1)

# statsmodels formula
import statsmodels.formula.api as smf
smf.ols(formula='wspeed ~ population', data=df)

# statsmodels OLS works for any degree polynomial
from sklearn.preprocessing import PolynomialFeatures
polynomial_features = PolynomialFeatures(degree=1)
xp1 = polynomial_features.fit_transform(x)
model = sm.OLS(y[:10], xp1[:10]).fit()

# sklearn
from sklearn.linear_model import LinearRegression
lm = LinearRegression().fit(X_train, y_train, sample_weight=None)
```



We are trying to find the "best" line that goes through the data... but "best" is a judgment call

minimize *something*:

$$\text{Sum of quared errors} : SSE = \sum_{i=0}^{N} (y_i - y_{i,predicted})^2$$

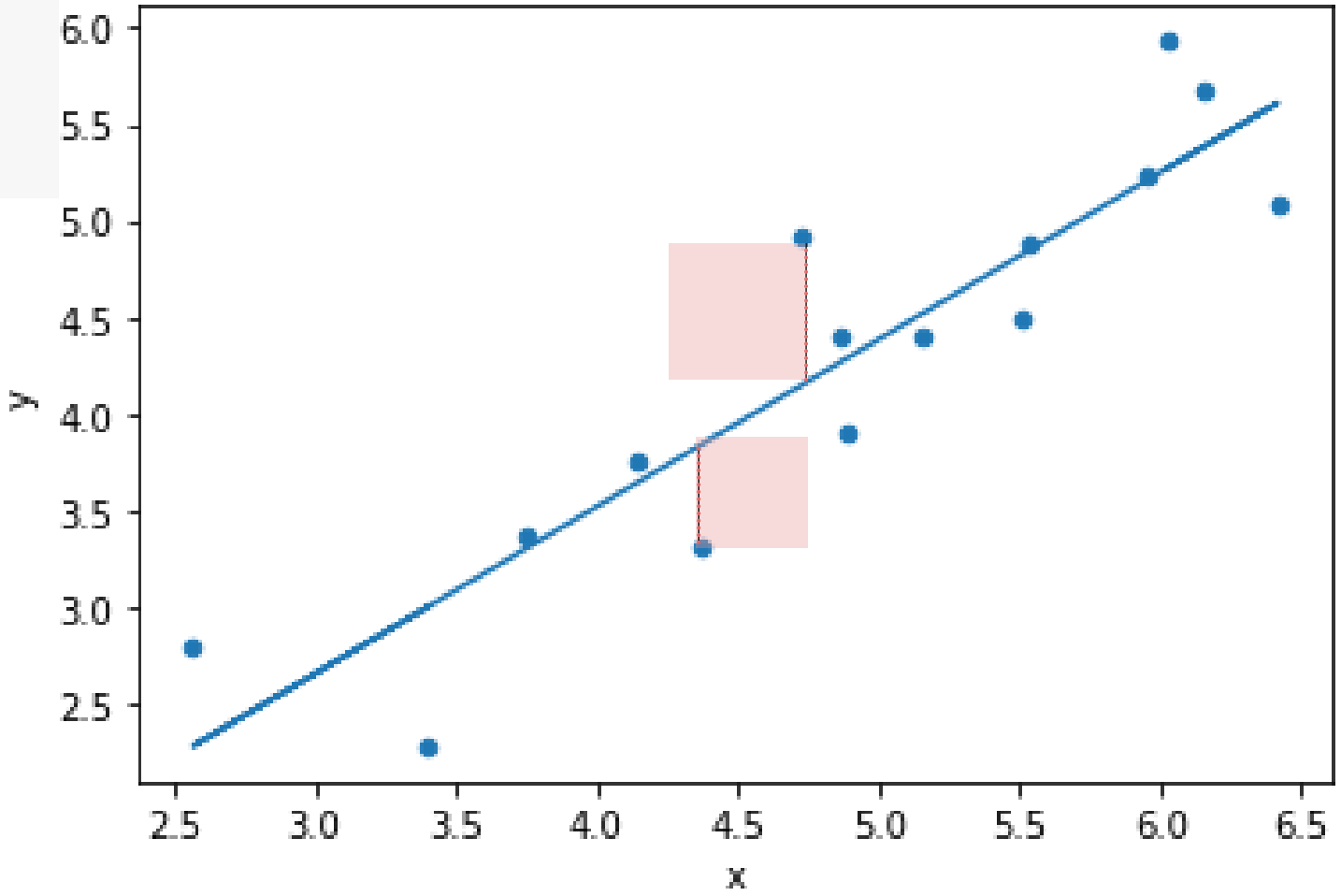
```
mod = smf.ols(formula='wspeed ~ population', data=df)

res = mod.fit()

res.summary()
```

OLS Regression Results

Dep. Variable:	wspeed	**R-squared:**	0.822
Model:	OLS	**Adj. R-squared:**	0.808
Method:	Least Squares	**F-statistic:**	59.90
Date:	Wed, 30 Sep 2020	**Prob (F-statistic):**	3.21e-06
Time:	13:06:28	**Log-Likelihood:**	-8.5829
No. Observations:	15	**AIC:**	21.17
Df Residuals:	13	**BIC:**	22.58
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	0.0566	0.560	0.101	0.921	-1.154	1.267
population	0.8653	0.112	7.740	0.000	0.624	1.107

Omnibus:	0.456	**Durbin-Watson:**	2.120
Prob(Omnibus):	0.796	**Jarque-Bera (JB):**	0.531
Skew:	0.115	**Prob(JB):**	0.767
Kurtosis:	2.107	**Cond. No.**	24.5



how good is the model?

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y}_i)^2}$$

← SSE ($\hat{y}_i$ prediction)

← variance ($\bar{y}_i$ mean)

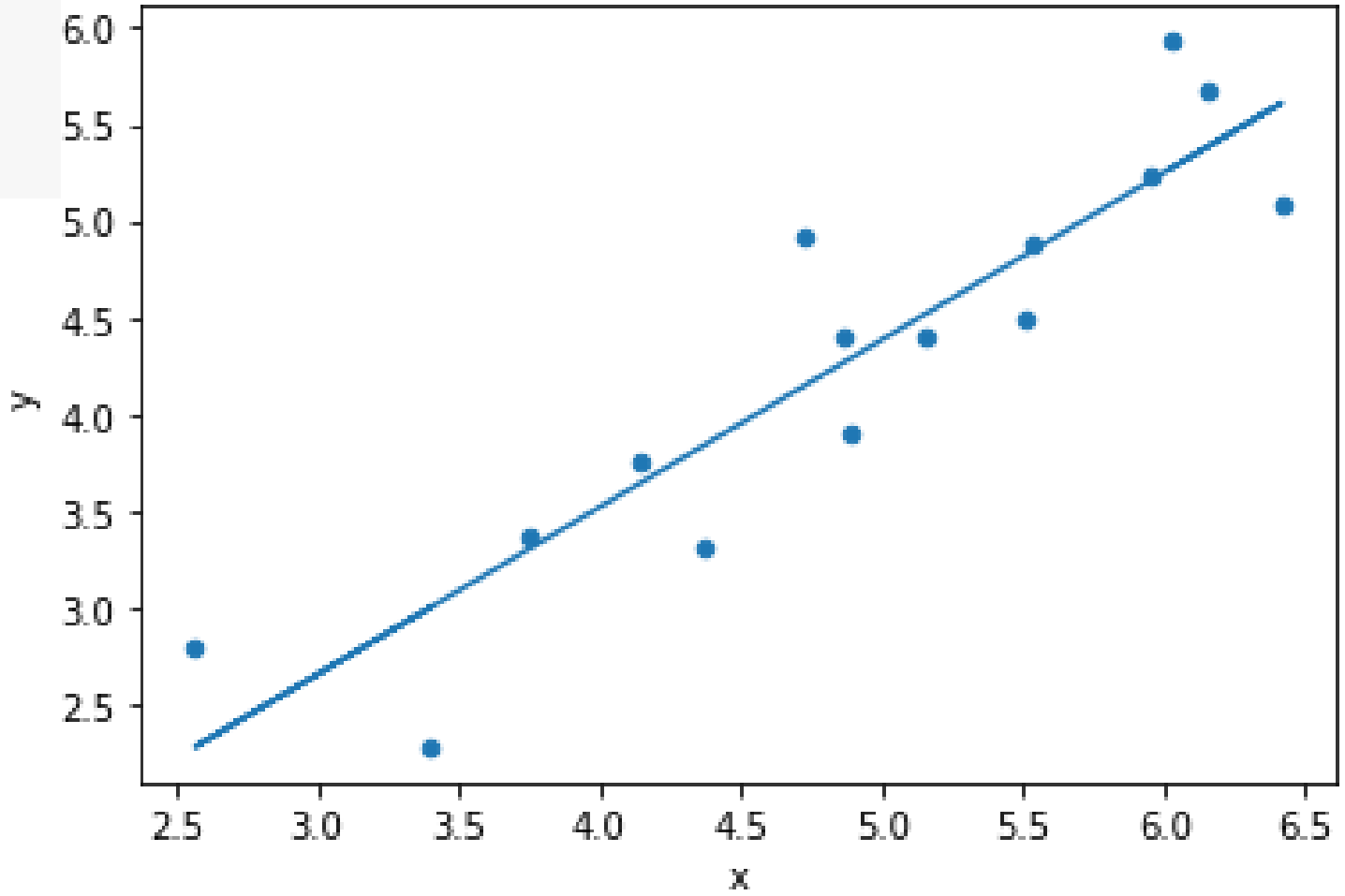
```
mod = smf.ols(formula='wspeed ~ population', data=df)

res = mod.fit()

res.summary()
```

OLS Regression Results

Dep. Variable:	wspeed	**R-squared:**	0.822
Model:	OLS	**Adj. R-squared:**	0.808
Method:	Least Squares	**F-statistic:**	59.90
Date:	Wed, 30 Sep 2020	**Prob (F-statistic):**	3.21e-06
Time:	13:06:28	**Log-Likelihood:**	-8.5829
No. Observations:	15	**AIC:**	21.17
Df Residuals:	13	**BIC:**	22.58
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	0.0566	0.560	0.101	0.921	-1.154	1.267
population	0.8653	0.112	7.740	0.000	0.624	1.107

Omnibus:	0.456	**Durbin-Watson:**	2.120
Prob(Omnibus):	0.796	**Jarque-Bera (JB):**	0.531
Skew:	0.115	**Prob(JB):**	0.767
Kurtosis:	2.107	**Cond. No.**	24.5



how good is the model?

$$R^2 adjusted = 1 - \frac{(1 - R^2)(N - 1)}{N - p - 1}$$

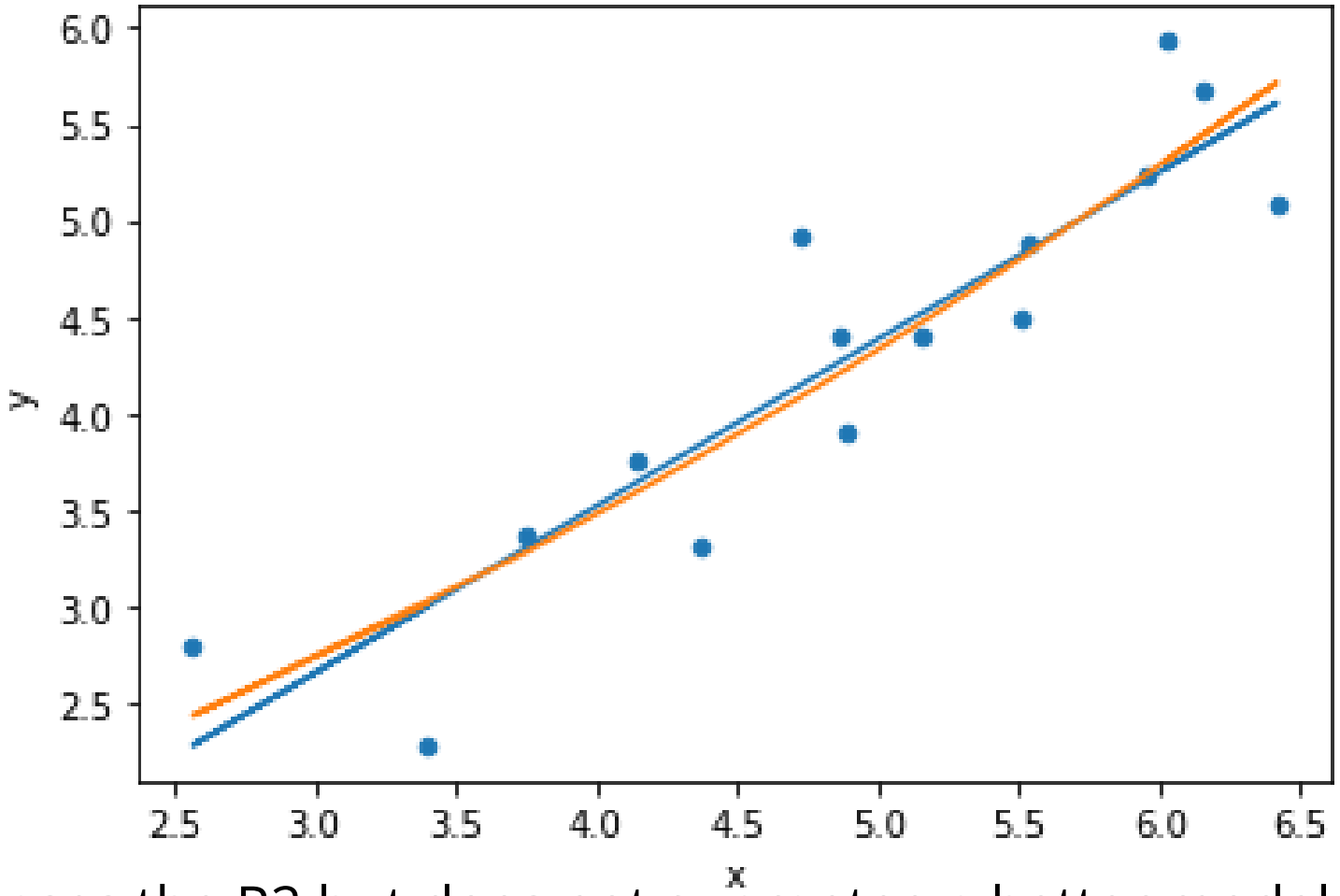
Adjusted R2 takes into account how many *datapoints* you have and how many *paramters* you have

```
mod = smf.ols(formula='wspeed ~ I(population**2) + population',
              data=df)
res = mod.fit()

res.summary()
```

Dep. Variable:	wspeed	**R-squared:**	0.826
Model:	OLS	**Adj. R-squared:**	0.797
Method:	Least Squares	**F-statistic:**	28.41
Date:	Wed, 30 Sep 2020	**Prob (F-statistic):**	2.81e-05
Time:	13:19:50	**Log-Likelihood:**	-8.4158
No. Observations:	15	**AIC:**	22.83
Df Residuals:	12	**BIC:**	24.96
Df Model:	2		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	1.0889	2.067	0.527	0.608	-3.415	5.593
I(population ** 2)	0.0513	0.099	0.520	0.613	-0.164	0.266
population	0.3916	0.918	0.426	0.677	-1.609	2.392

Omnibus:	0.026	**Durbin-Watson:**	2.071
Prob(Omnibus):	0.987	**Jarque-Bera (JB):**	0.238
Skew:	0.051	**Prob(JB):**	0.888
Kurtosis:	2.392	**Cond. No.**	507.



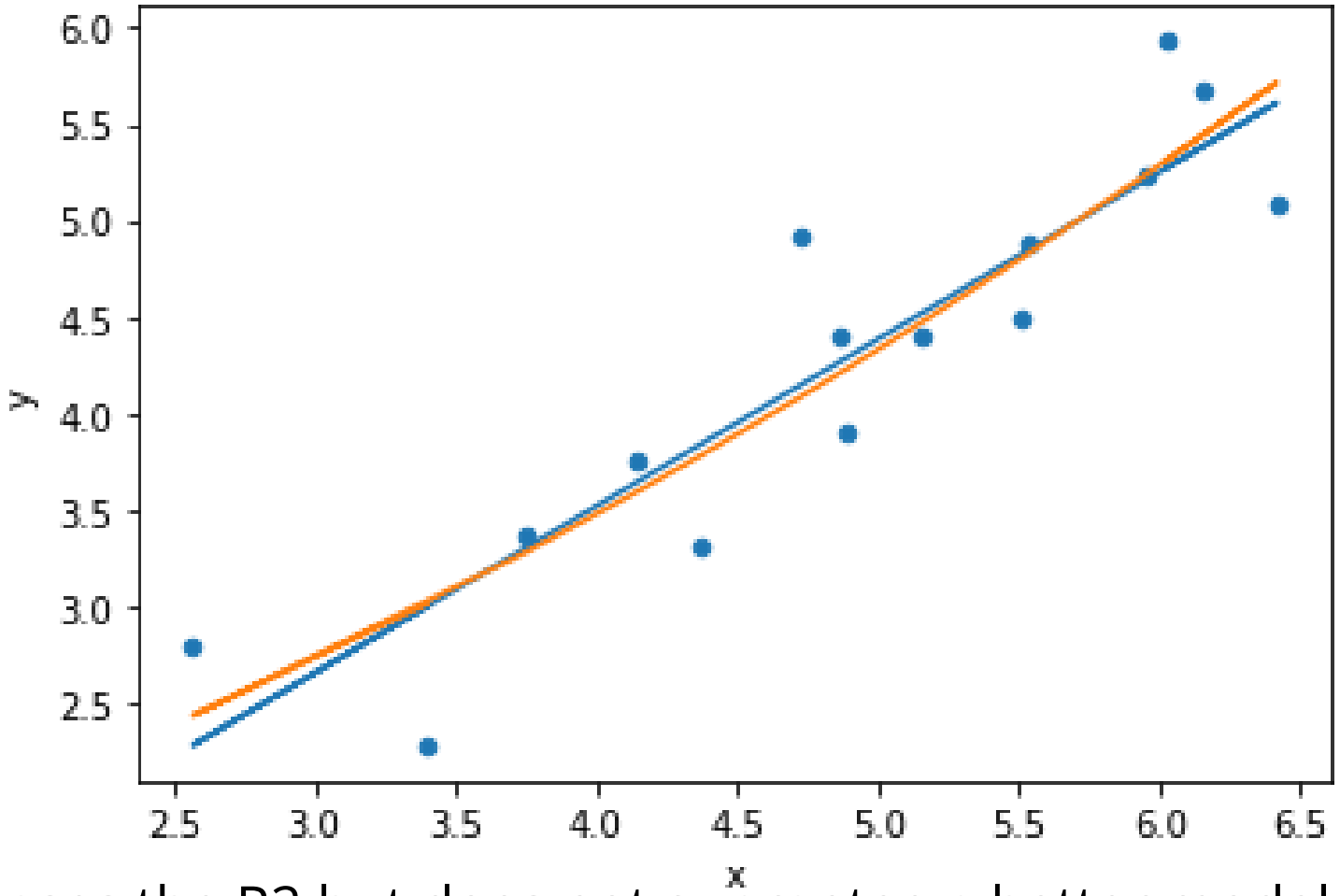
- increasing the model complexity increases the R2 but does not guarantee a better model
- the likelihood ratio is a way to assess if the more complex model is better in a NHRT sense

```
mod = smf.ols(formula='wspeed ~ I(population**2) + population',
              data=df)
res = mod.fit()

res.summary()
```

Dep. Variable:	wspeed	**R-squared:**	0.826
Model:	OLS	**Adj. R-squared:**	0.797
Method:	Least Squares	**F-statistic:**	28.41
Date:	Wed, 30 Sep 2020	**Prob (F-statistic):**	2.81e-05
Time:	13:19:50	**Log-Likelihood:**	-8.4158
No. Observations:	15	**AIC:**	22.83
Df Residuals:	12	**BIC:**	24.96
Df Model:	2		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	1.0889	2.067	0.527	0.608	-3.415	5.593
I(population ** 2)	0.0513	0.099	0.520	0.613	-0.164	0.266
population	0.3916	0.918	0.426	0.677	-1.609	2.392

Omnibus:	0.026	**Durbin-Watson:**	2.071
Prob(Omnibus):	0.987	**Jarque-Bera (JB):**	0.238
Skew:	0.051	**Prob(JB):**	0.888
Kurtosis:	2.392	**Cond. No.**	507.



- increasing the model complexity increases the R2 but does not guarantee a better model
- the likelihood ratio is a way to assess if the more complex model is better in a NHRT sense

NH: the more complex model is better

under the NH the LR statistics is distributed like a chi^2 distribution with number of dof = number of extra parameters

p-value of LR in a chi^2 distribution with dof

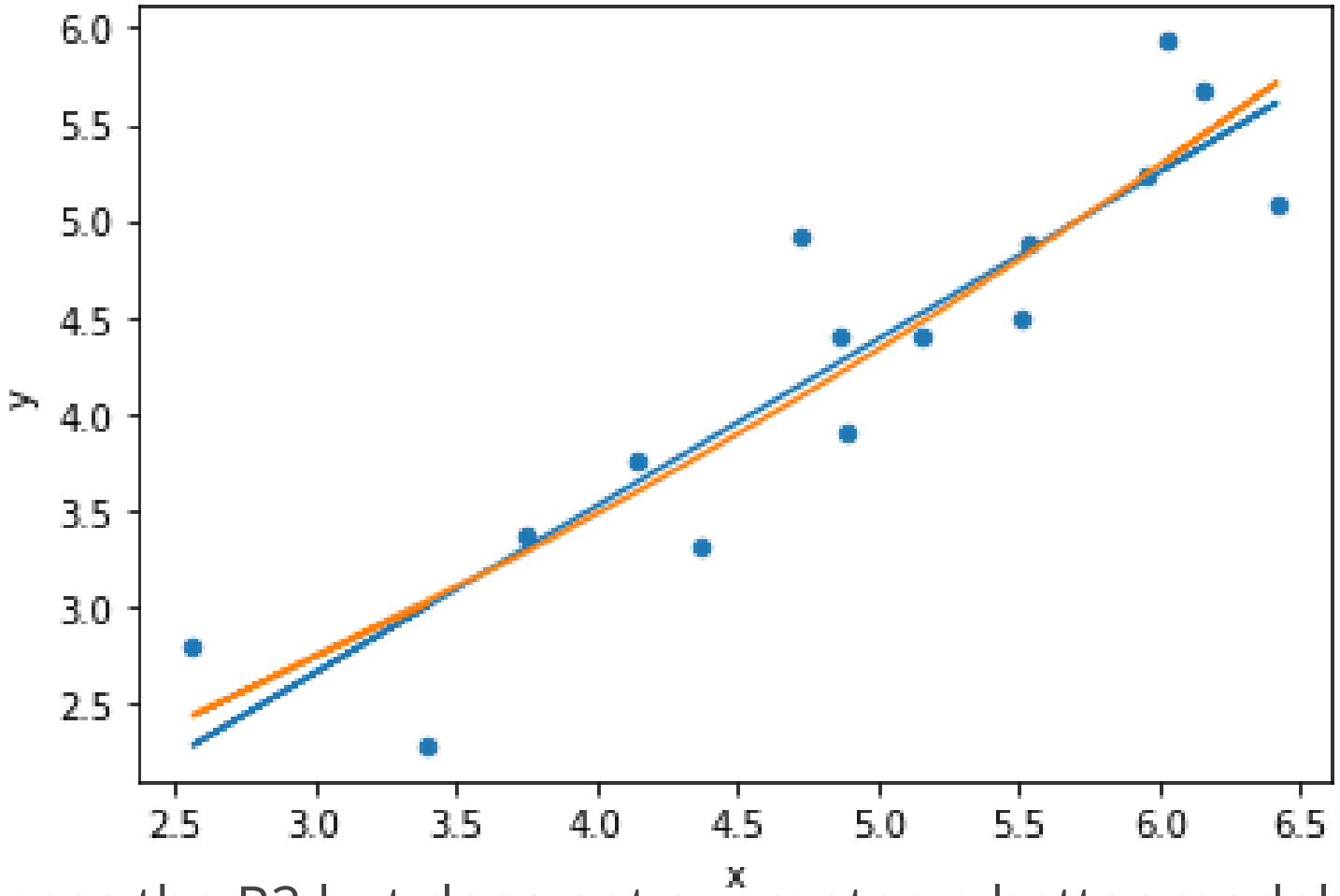
```
res2.compare_lr_test(res)
(0.3342540144461843, 0.5631648421666332, 1.0)
```

```
mod = smf.ols(formula='wspeed ~ I(population**2) + population',
              data=df)
res = mod.fit()

res.summary()
```

Dep. Variable:	wspeed	**R-squared:**	0.826
Model:	OLS	**Adj. R-squared:**	0.797
Method:	Least Squares	**F-statistic:**	28.41
Date:	Wed, 30 Sep 2020	**Prob (F-statistic):**	2.81e-05
Time:	13:19:50	**Log-Likelihood:**	-8.4158
No. Observations:	15	**AIC:**	22.83
Df Residuals:	12	**BIC:**	24.96
Df Model:	2		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	1.0889	2.067	0.527	0.608	-3.415	5.593
I(population ** 2)	0.0513	0.099	0.520	0.613	-0.164	0.266
population	0.3916	0.918	0.426	0.677	-1.609	2.392

Omnibus:	0.026	**Durbin-Watson:**	2.071
Prob(Omnibus):	0.987	**Jarque-Bera (JB):**	0.238
Skew:	0.051	**Prob(JB):**	0.888
Kurtosis:	2.392	**Cond. No.**	507.



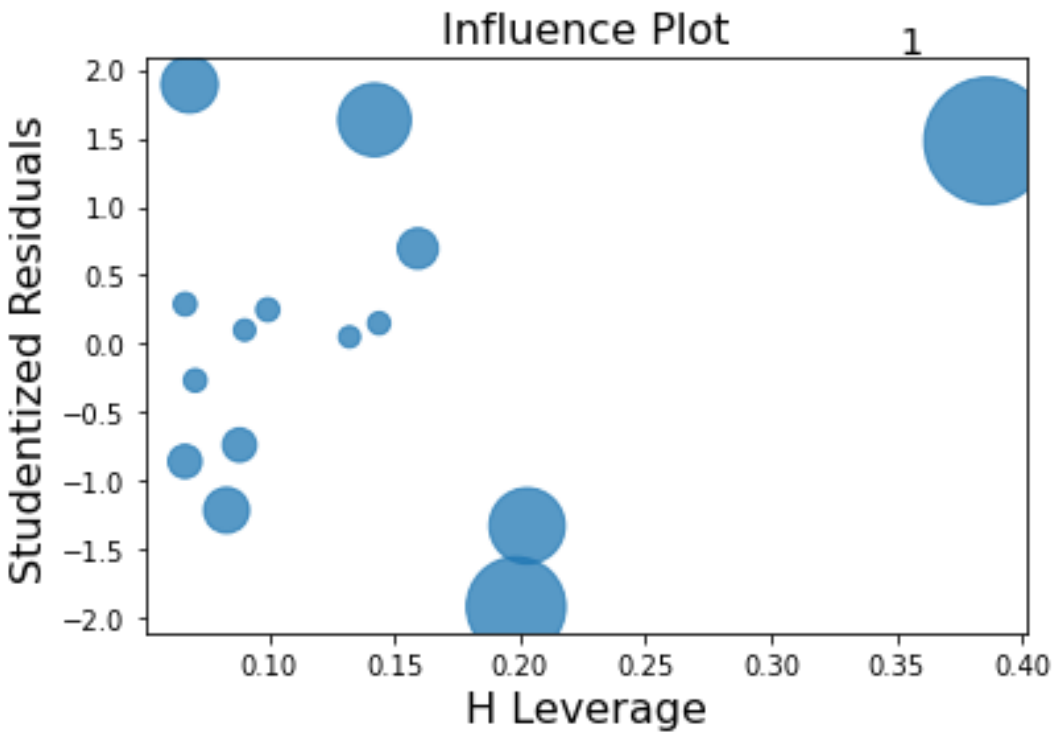
- increasing the model complexity increases the R2 but does not guarantee a better model
- the likelihood ratio is a way to assess if the more complex model is better in a NHRT sense

```
likelihood ration test - set alpha to alpha=0.05

 the NH is that the more complex model is better than the simpler one

 the likelihood ration statistics is 0.33, which corresponds to a p-value of 0.56

since the likelishood ration statistics is chi square distributed
with DoF the difference in the number of parameters in the 2 models (=1 here)
this corresponds not being able to reject the NH at alpha 0.05
```

```
res2.compare_lr_test(res)
```

```
(0.3342540144461843, 0.5631648421666332, 1.0)
```

not all points are equal!

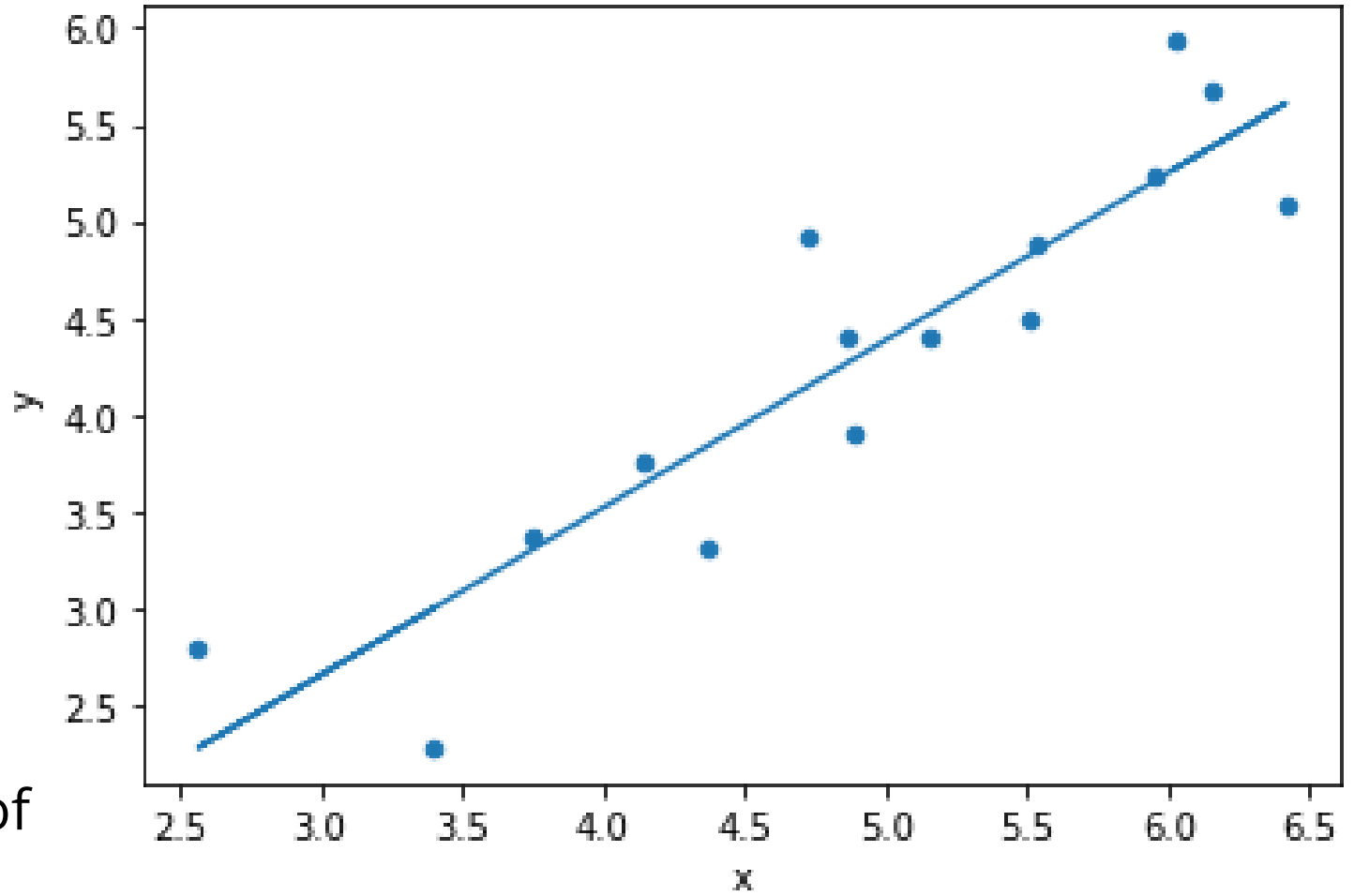


This function creates a “bubble” plot of Studentized residuals versus hat leverage values.

hi residual: outlier on the Y axis

high leverage : at the edge of the X axis

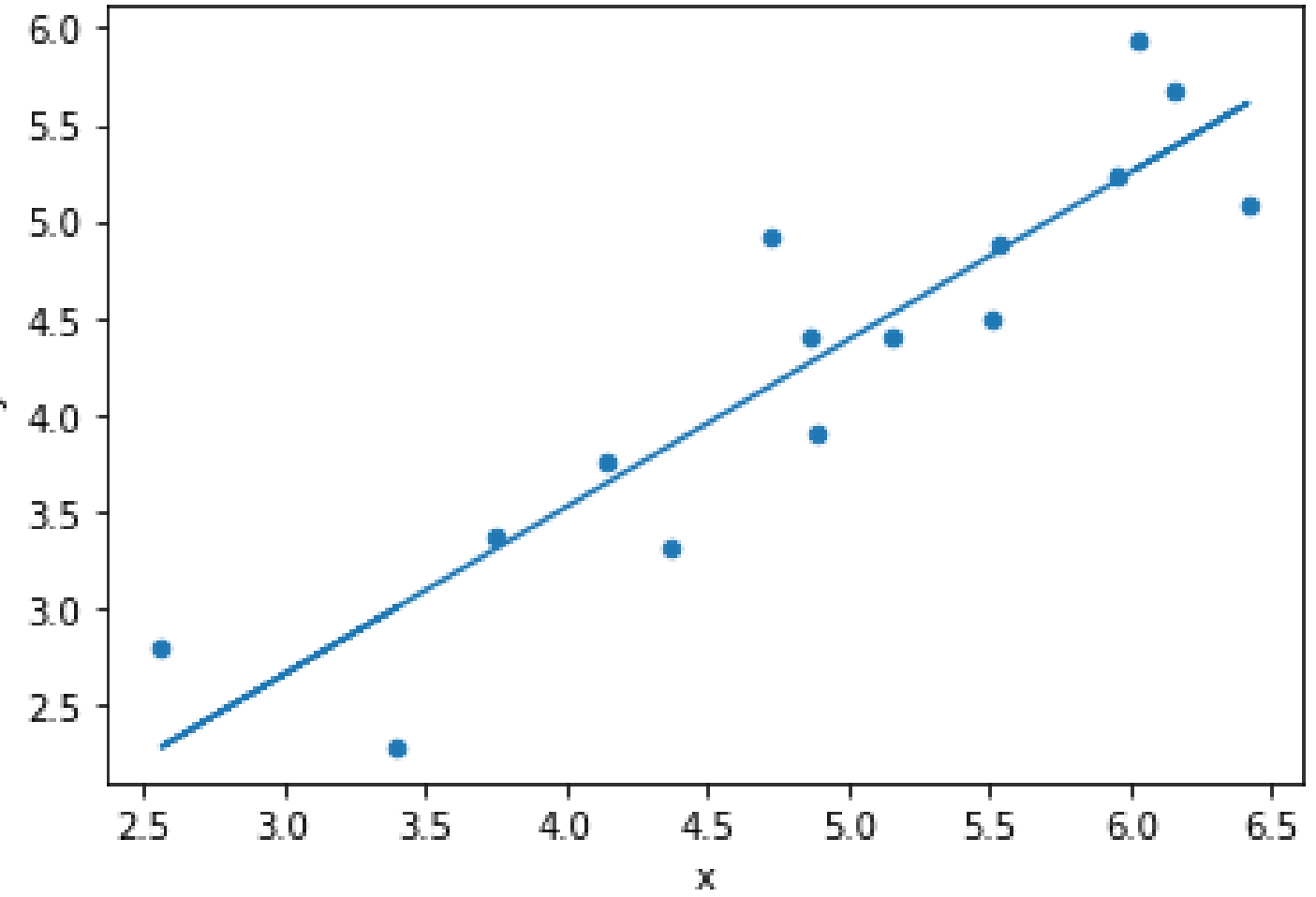
influence analysis



identify data points that have strong influence on the model fit: unusual *x* values or their *y* value is an "outlier". Points that are both are on the top right of the plot and are **high influence points**. Cook's distance is represented by the size of the bubble.

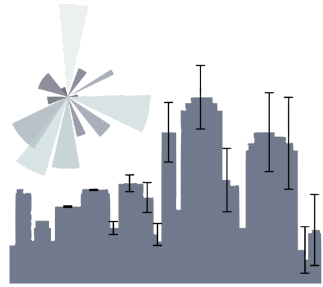
Normal equation

It can be shown that OLS minimization of the sum of the square errors (SSE) is equivalent to calculating the slope and intercept as:



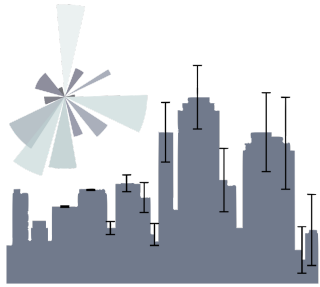
$$s = \frac{\sum_{i=1}^{N}(x_i - \bar{X})(y_i - \bar{Y})}{\sum_{i=1} N(x_i - \bar{X})^2}$$

$$b = \bar{Y} - \bar{X}$$



5 how to avoid overfitting in machine learning

Cross validation



Train and Test split

```
[46] from sklearn.linear_model import LinearRegression
     from sklearn.model_selection import train_test_split
     #  split train and test set
     X_train, X_test, y_train, y_test = train_test_split(
       x, y, test_size=0.33, random_state=302)
```

```
[47] # fit to training data
     lm = LinearRegression().fit(X_train,
                                 y_train, sample_weight=None)
```

```
[50] # in sample score
     print("linear regression in-sample score: {:.0f}% ".format(
         lm.score(X_train,  y_train) * 100))
```

```
linear regression in-sample score: 88%
```

```
# out of sample score
print("linear regression ot-of-sample score: {:.0f}% ".format(
    lm.score(X_test,  y_test) * 100))
```

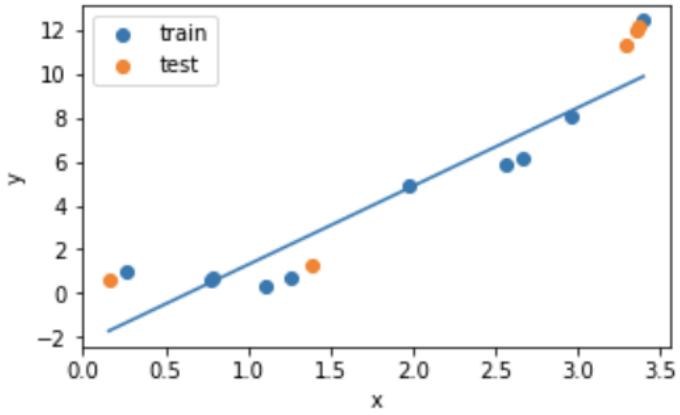
```
linear regression ot-of-sample score: 85%
```

poor model: - low in-sample score

overfitting: - high in-sample score, low out-of-sample score

a small drop in score, like the one seen above, is normal

```
plt.figure(figsize=(5,3))
plt.scatter(X_train, y_train, label="train")
plt.scatter(X_test, y_test, label="test")
plt.plot(x, lm.predict(x))
plt.xlabel('x')
plt.ylabel('y')
plt.legend();
```



key concepts

geopandas

line models and polynomial models

parameters and hyperparameters

objective function: what do we minimize to choose parameters?

model diagnostics and choosing models

influence points

cross validation

references

this is a pretty comprehensive and clear medium post with coding examples

https://medium.com/@kylecaron/introduction-to-linear-regression-part-1-implementation-in-python-with-statsmodels-7dbf24461072

this is a whole paper about modeling in practice and in theory which covers extensively how to deal with uncertainties. it is not for the faint of heart. The notes are entertaining

https://arxiv.org/abs/1008.4686

https://www.nature.com/articles/nmeth.3812.pdf

POINTS OF SIGNIFICANCE

Analyzing outliers: influential or nuisance?

Some outliers influence the regression fit more than others.

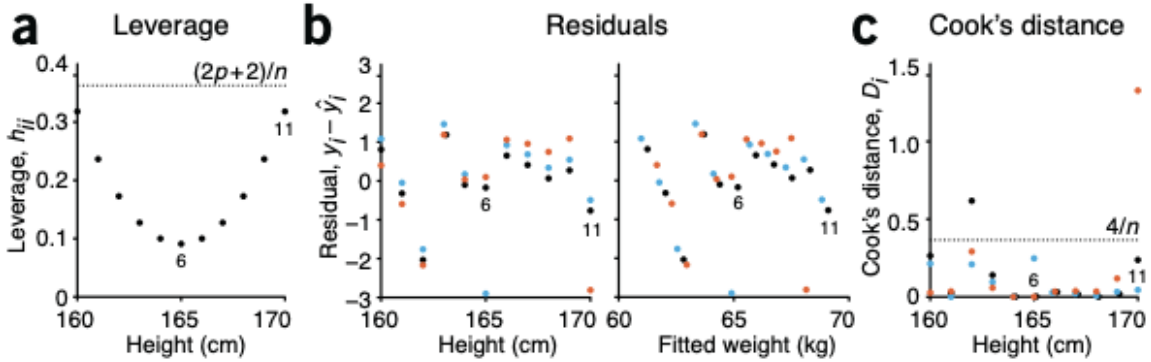
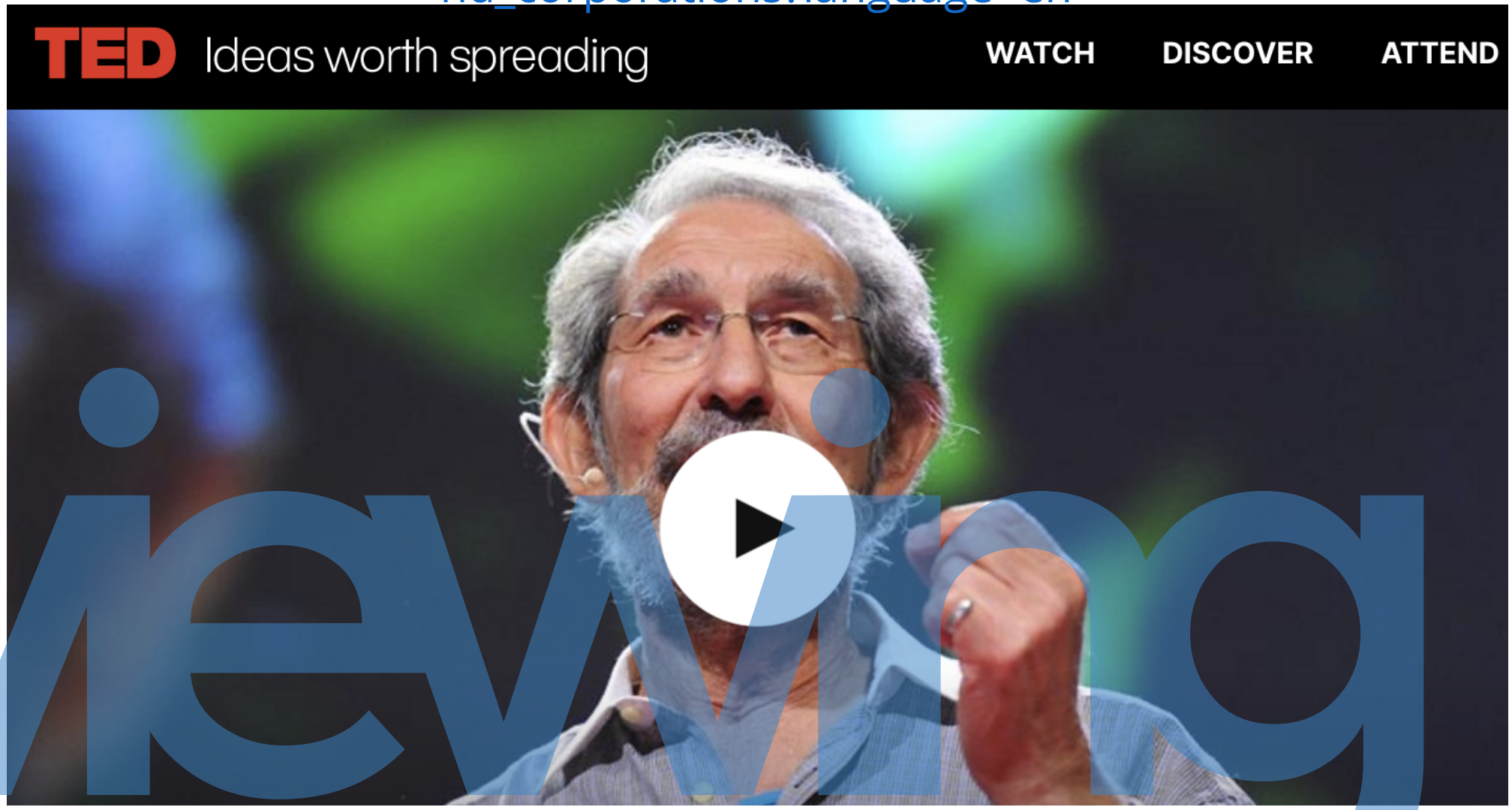


Figure 2 | The leverage, residual and Cook's distance of an observation are used to assess the robustness of the fit. (**a**) The leverage of an observation tells us about its potential to influence the fit and increases as the square

https://www.ted.com/talks/geoffrey_west_the_surprising_math_of_cities_and_corporations?language=en



viewing

homework