Machine learning in glass science, with examples of applications

Charles Le Losq

Géomatériaux, Institut de physique du globe de Paris

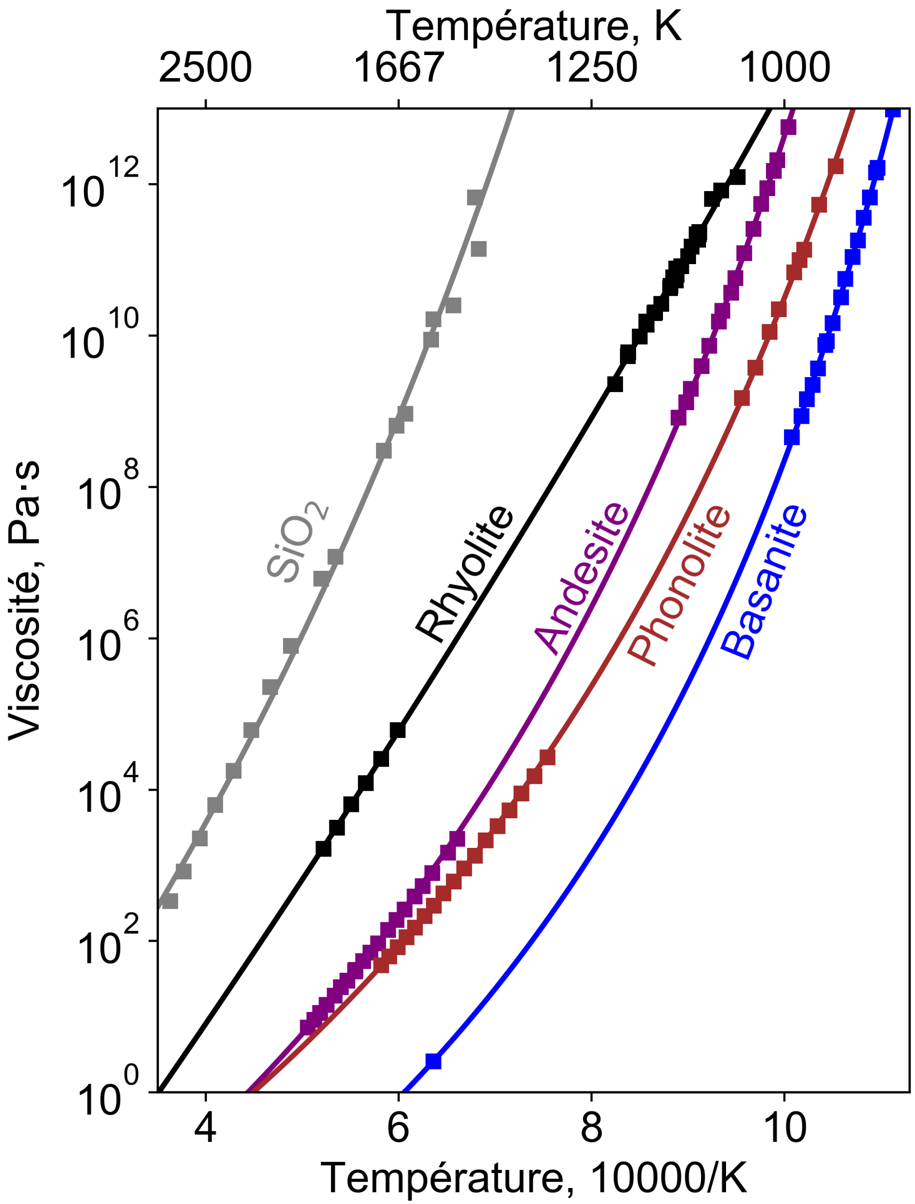
Collaborations:

IPGP: Daniel Neuville, Roberto Moretti
ANU-RSES: Matt Valetich, Richard Arculus, John Mavrogenes, Hugh O'Neill, Andrew Berry, Malcolm Sambridge
Durham University: Andrew Valentine
Geophysical Lab: Bjorn Mysen, George Cody

Merci à l'ANR IDEX Université de Paris, 18-IDEX-0001

Jounées USTV 2021

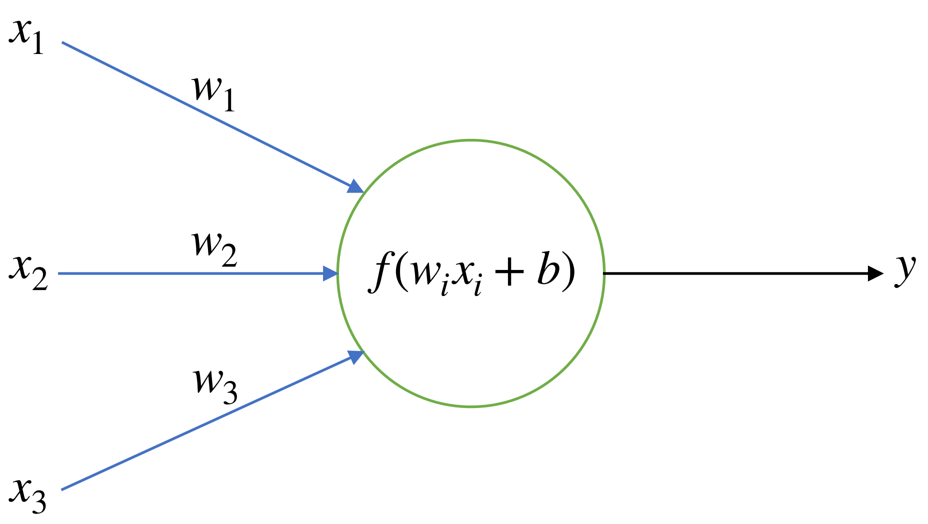
The viscosity of silicate melts



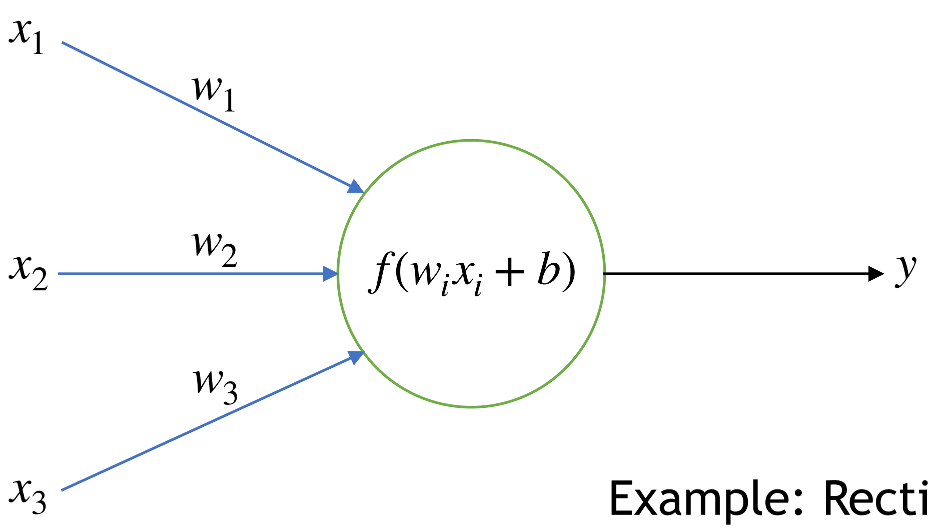
- Chemistry > structure > properties
- Volatiles, iron redox…
- Also crystals and bubbles

No general (precise) model that bring a true understanding of the flow process

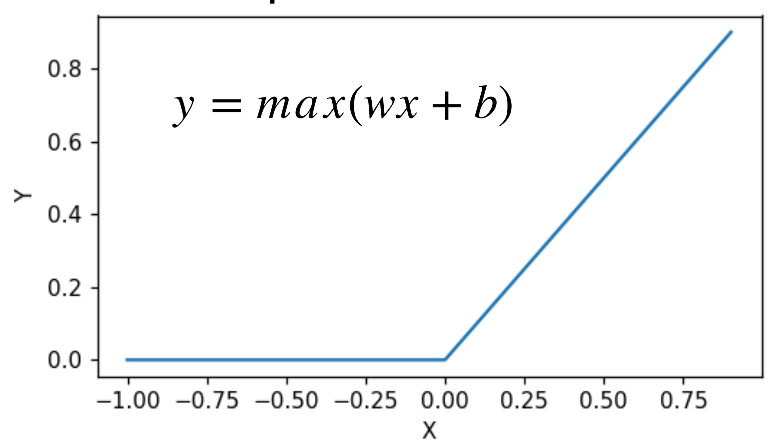
Perceptron Model



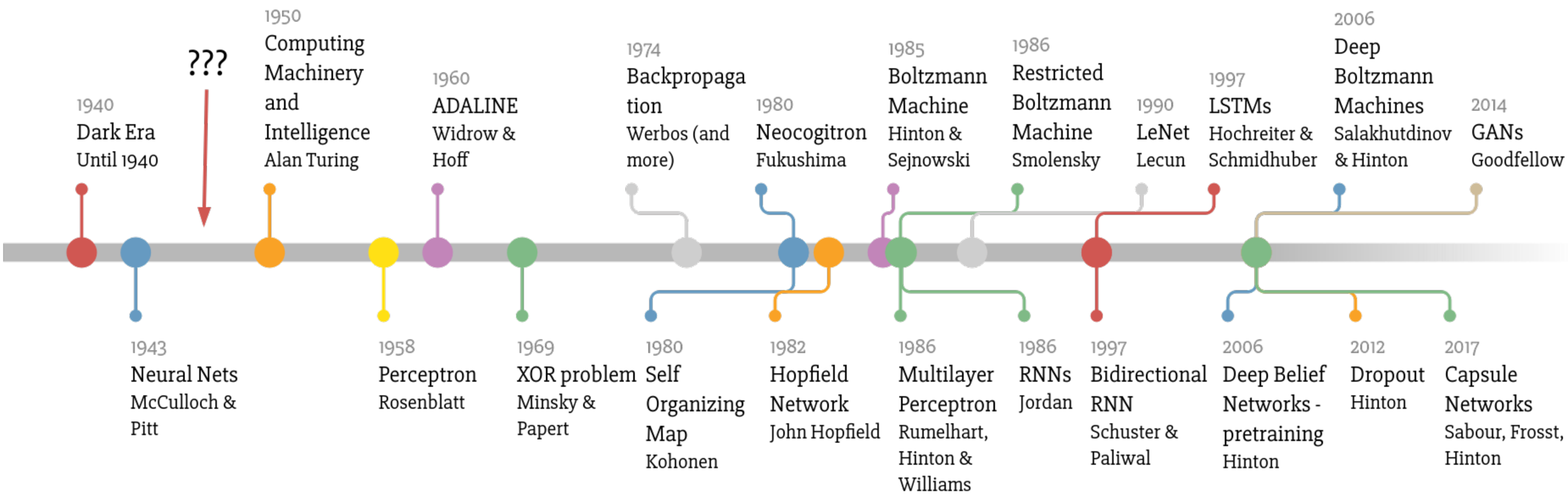
Perceptron Model



Example: Rectifier function



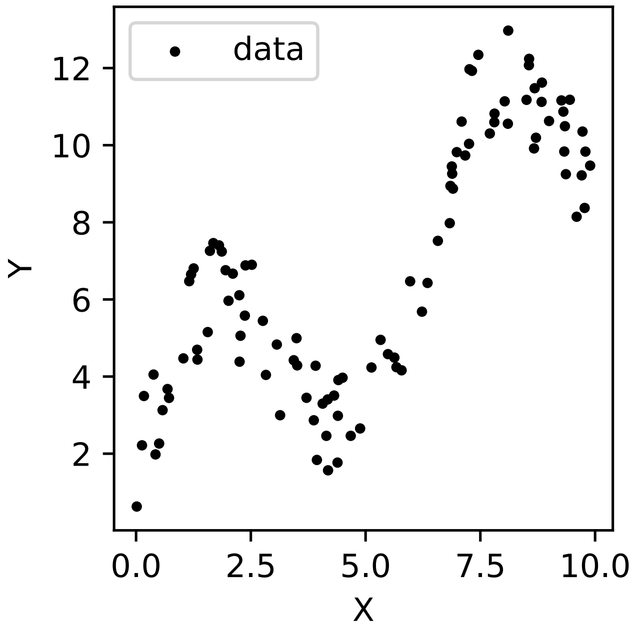
Deep Learning Timeline



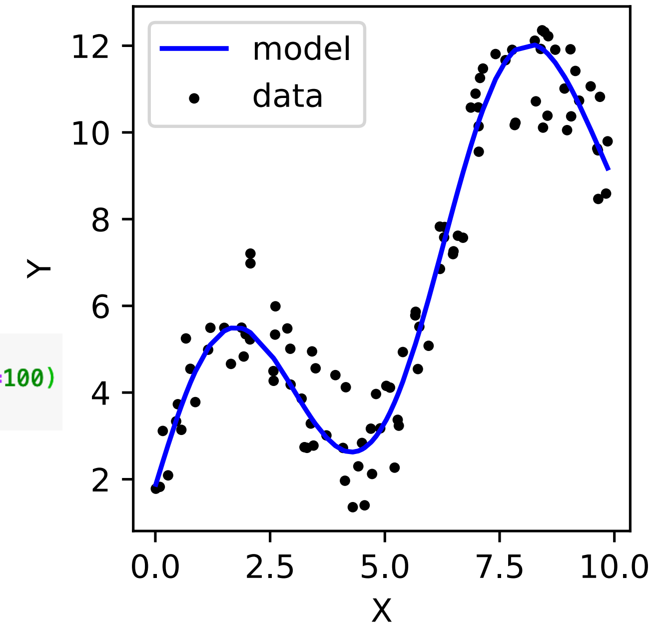
Made by Favio Vázquez

Why « trendy »?

- It’s now easy = focus on building models, not code
  - open source software libraries
  - high level languages (Julia, Python, R, Matlab)




```python
from sklearn.svm import SVR
svr_rbf = SVR(kernel="rbf", C=100)
svr_rbf.fit(x.reshape(-1,1),y)
```



- Internet
  - amount and transfers of data
  - easy access to resources
- Social media / advertisement driven
  - two of the largest ML software libraries, Tensorflow (2015) and Pytorch (2016), « surprizingly » originate from Google and Facebook

What can it bring to us?

Glass: structure & properties from measurements, theories and MD simulations

Machine learning = another tool in our toolbox to, e.g.,

- analyze data and generate models
- explore datasets
- design new compositions
- detect drifts & anomalies

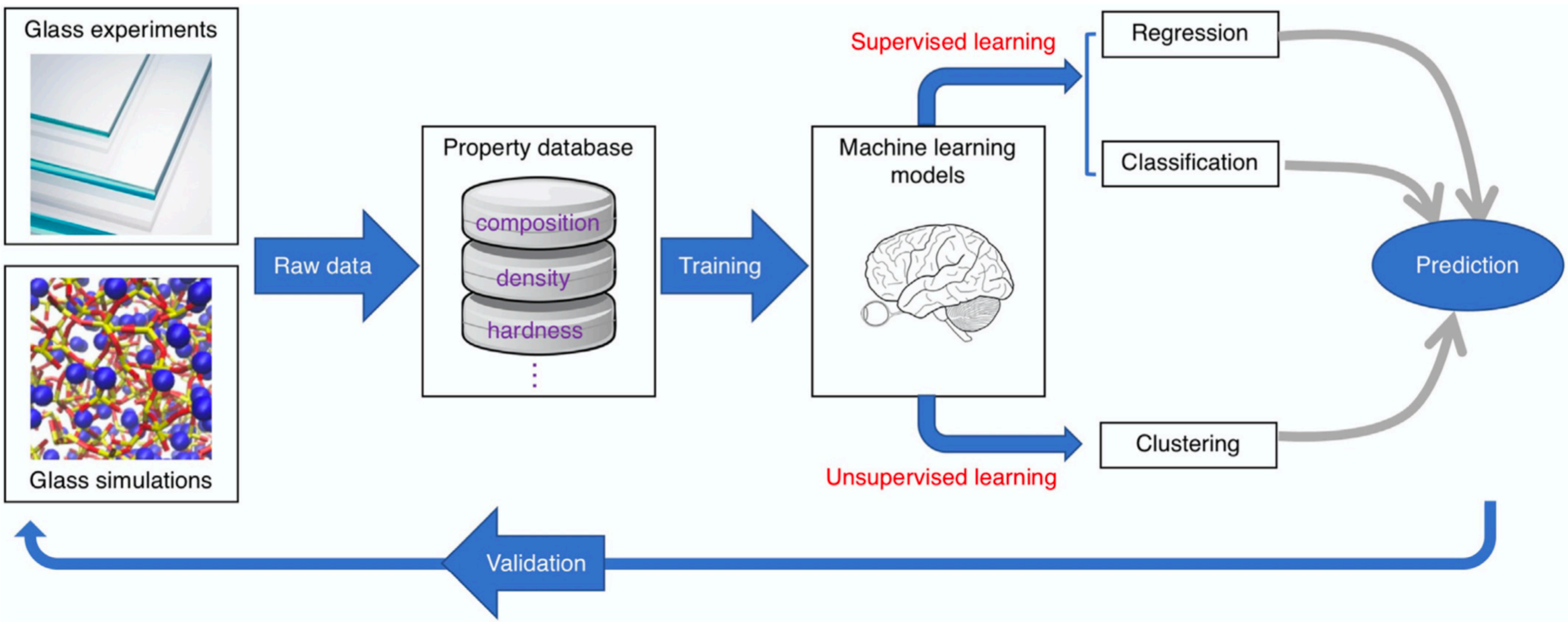
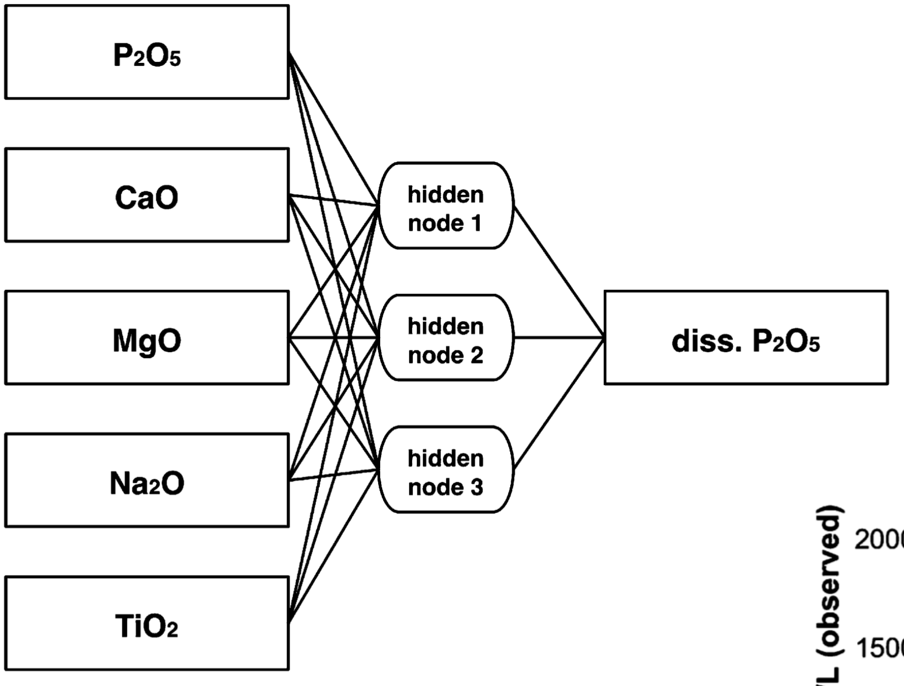


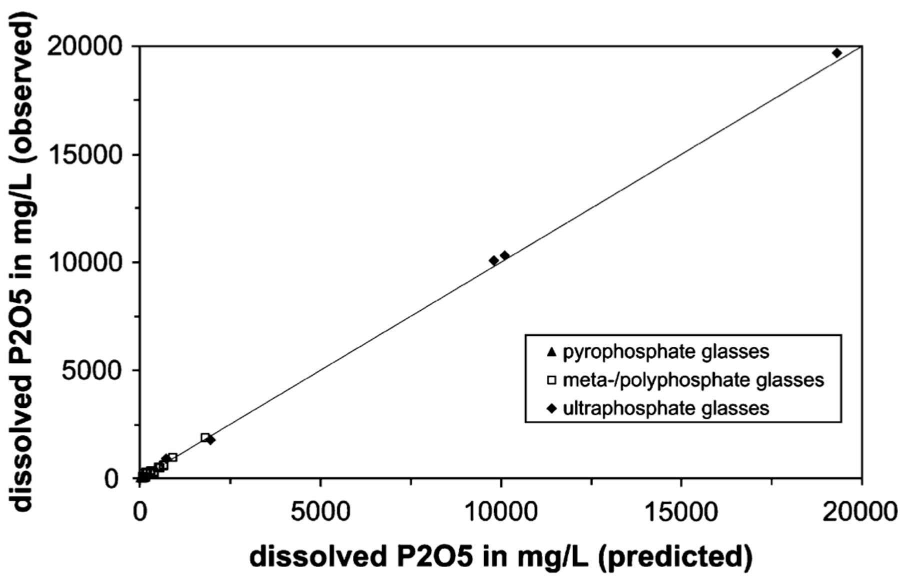
Figure Liu et al. 2019

Examples of applications

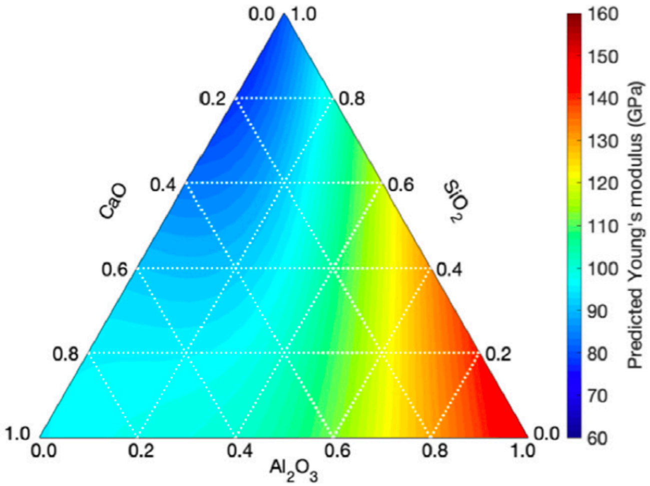


Brauer et al. 2007

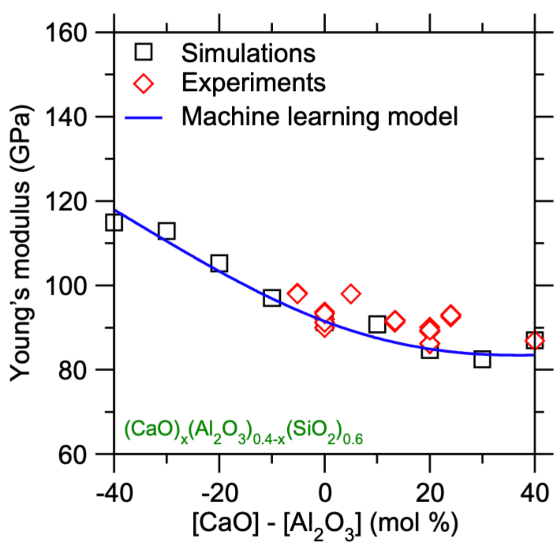
Degradable implant materials: solubility of glasses in the system P_2O_5-CaO-MgO-N



Examples of applications



(b)



(d)

Yang et al. 2019

Predictions of mechanical properties of glasses

Youngs modulus in CaO aluminosilicates

Practical example: how to predict glass density from composition?

Practical example: how to predict glass density from composition?

Prediction of the density of Na_2O-K_2O-Al_2O_3-SiO_2 glasses

Supervised ML: we know X and Y

- Practical example: prediction of the density of Na_2O-K_2O-Al_2O_3-SiO_2 glasses
    - Database available at github.com/charlesll/i-melt/

Database_IPGP.xlsx

Name	sio2	al2o3	na2o	k2o	d	ese_d	reference	Name_study
NAK83.8.0	83,9	8,7	7,4	0,0	2,317	0,005	LL2013	NAK83.8.0
NAK83.8.2	83,4	8,3	6,2	2,1	2,318	0,007	LL2013	NAK83.8.2
NAK83.8.4	83,2	8,4	4,2	4,1	2,317	0,004	LL2013	NAK83.8.4
NAK83.8.6	83,3	8,5	2,2	6,1	2,323	0,006	LL2013	NAK83.8.6
NAK83.8.8	83,1	8,6	0,1	8,2	2,313	0,007	LL2013	NAK83.8.8
KA8005	84,1	5,0	0,0	10,8	2,345	0,001	TS	KA8005
NAK75.0.25	75,0	0,0	0,0	25,0	2,426	0,005	LL2017b	NAK75.0.25
NAK75.0.0	75,0	0,0	25,0	0,0	2,430	0,004	LL2014	NAK75.0.0
NAK75.2.0	76,5	2,1	21,4	0,0	2,416	0,008	LL2014	NAK75.2.0
NAK75.06.0	75,2	6,4	18,4	0,0	2,402	0,001	LL2014	NAK75.06.0
NAK75.9.0	75,1	9,2	15,7	0,0	2,393	0,003	LL2014	NAK75.9.0
NAK75.12.0	75,0	12,5	12,5	0,0	2,369	0,001	LL2013	NAK75.12.0
NAK75.12.2	75,4	12,5	9,7	2,4	2,374	0,003	LL2013	NAK75.12.2
NAK75.12.5	75,3	12,4	7,4	4,8	2,368	0,003	LL2013	NAK75.12.5
NAK75.12.6	75,2	12,5	6,1	6,2	2,371	0,003	LL2013	NAK75.12.6
NAK75.12.7	75,5	12,3	5,0	7,2	2,372	0,001	LL2013	NAK75.12.7
NAK75.12.10	75,2	12,4	2,7	9,8	2,370	0,004	LL2013	NAK75.12.10
NAK75.12.12	75,1	12,8	0,1	12,1	2,357	0,008	LL2013	NAK75.12.12
NAK75.16.0	75,2	16,1	8,8	0,0	2,375	0,004	LL2014	NAK75.16.0
KA72.07	72,3	7,1	0,0	20,6	2,408	0,001	TS	KA72.07
NAK67.16.0	66,7	16,7	16,7	0,0	2,422	0,006	LL2017	NAK67.16.0
NAK67.16.16	66,7	16,7	0,0	16,7	2,401	0,005	LL2017	NAK67.16.16
NS66	66,7	0,0	33,3	0,0	2,483	0,001	N2006	NS66

1/ Data Importation

- **We import the data (Python, library Pandas)**

```
data = pd.read_excel("Database_IPGP.xlsx")

X = data.loc[:,["sio2","al2o3","na2o","k2o"]].values
Y = data.loc[:,"d"].values.reshape(-1,1)
XY = data.loc[:,["sio2","al2o3","na2o","k2o","d"]].values
```

$Y = f(\mathrm{X})$, f unknown

2/ Data Preparation

- We import the data (Python, library Pandas)
- **We prepare the data**
  - Split the data
    - train & validation subset(s) for training
    - test subset for final evaluation

```
# Splitting the data !
X_train_valid, X_test, Y_train_valid, Y_test = train_test_split(X, Y, test_size=0.30)
X_train, X_valid, Y_train, Y_valid = train_test_split(X_train_valid, Y_train_valid, test_size=0.20)
```

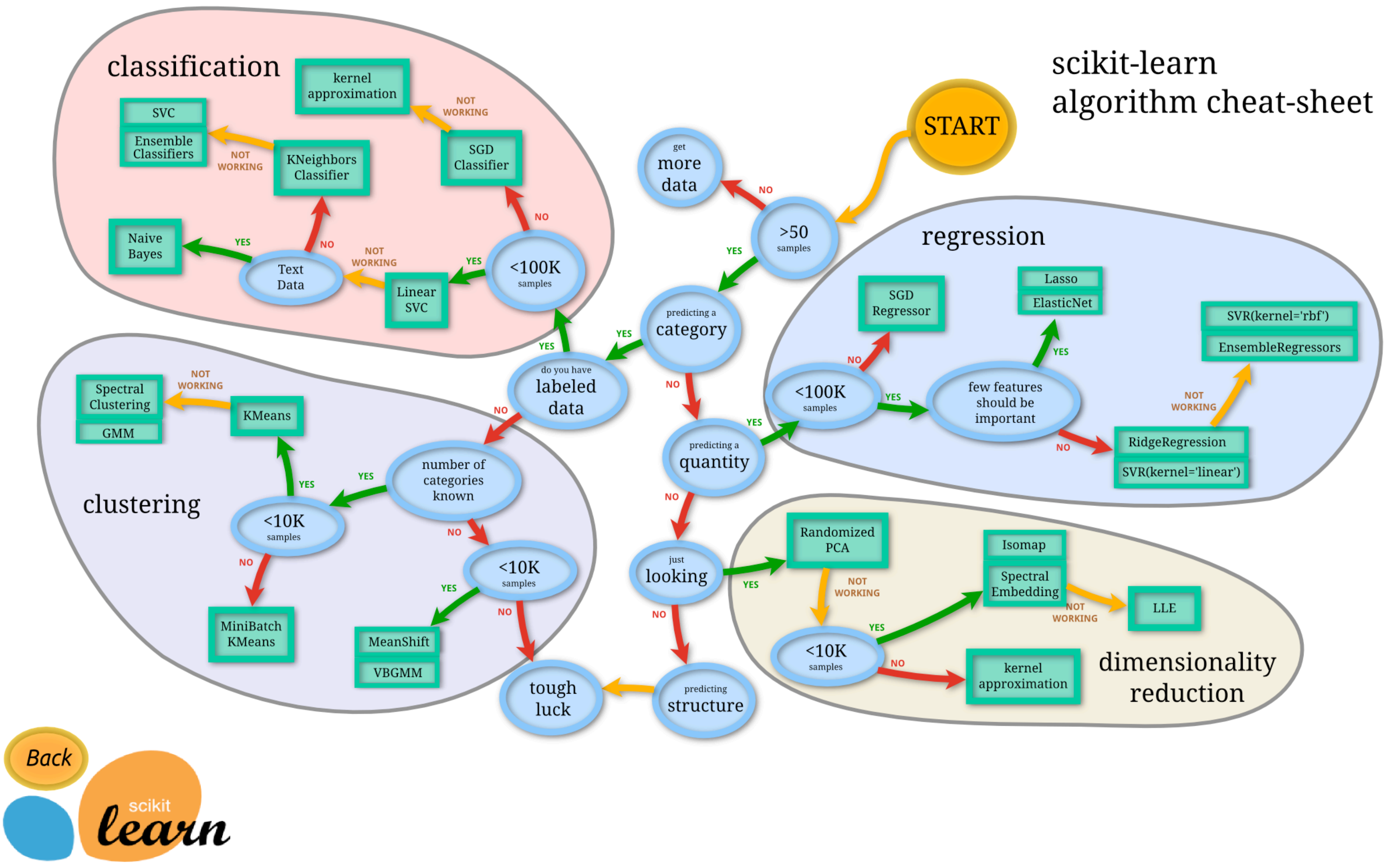
  - Standardize the data = ML algorithms behave better with numbers at the same dimension, and usually close to unity
    - min-max = *(X-min(X))/(max(X)-min(X))*
    - standard scaler = *(X-mean(X))/std(X)*

```
#### Scaling X data : the composition
X_scaler = StandardScaler() # creating the scaler
# do the job
X_train_sc = X_scaler.fit_transform(X_train)
X_valid_sc = X_scaler.fit_transform(X_valid)
X_test_sc = X_scaler.fit_transform(X_test)

#### Scaling the Y data
Y_scaler = StandardScaler() # creating the scaler
# do the job
Y_train_sc = Y_scaler.fit_transform(Y_train)
Y_valid_sc = Y_scaler.fit_transform(Y_valid)
Y_test_sc = Y_scaler.fit_transform(Y_test)

#### Scaling the full dataset
XY_sc= X_scaler.fit_transform(XY)
```

4/ Model selection

- We import the data (Python, library Pandas)
- We prepare the data
  - Split the data
    - train & validation subset(s) for training
    - test subset for final evaluation
  - Standardize the data = ML algorithms behave better with numbers at the same dimension, and usually close to unity
    - min-max = *(X-min(X))/(max(X)-min(X))*
    - standard scaler = *(X-mean(X))/std(X)*
- **We need to choose a model**

4/ Model selection

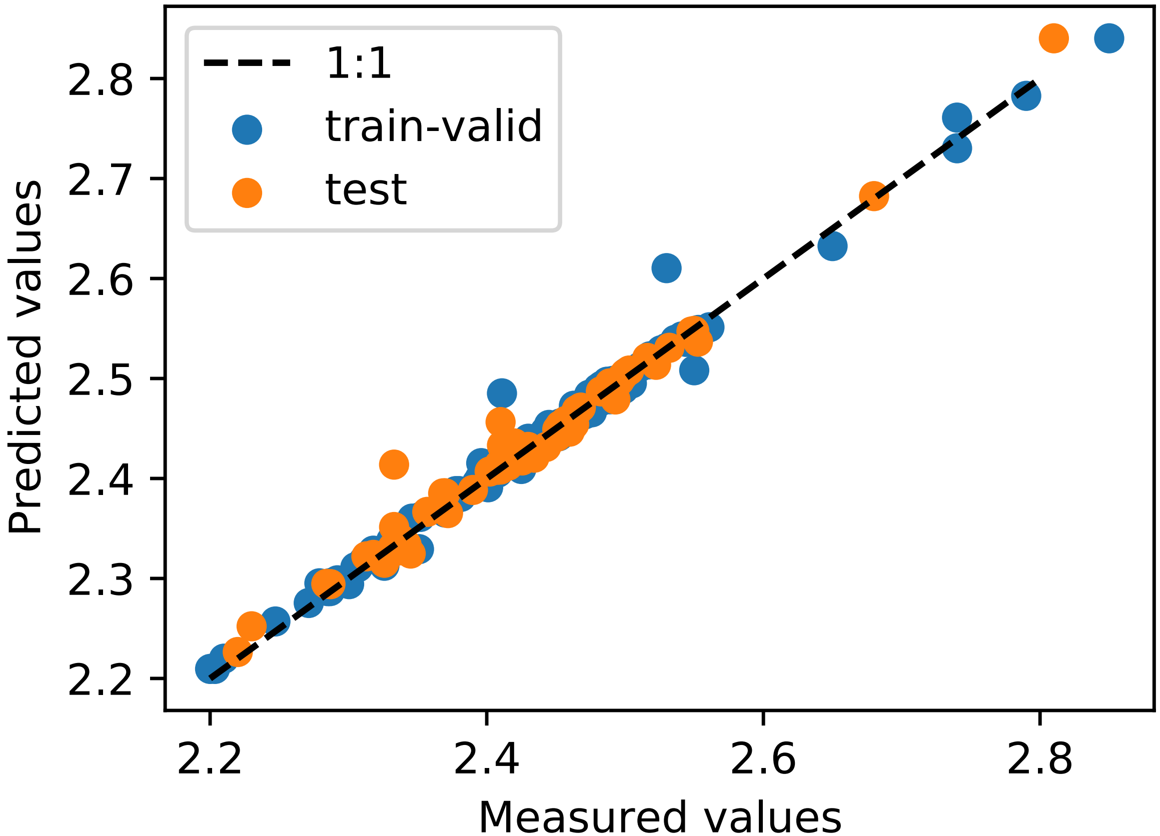


5/ Fit and predict!

- We import the data (Python, library Pandas)
- We prepare the data
  - Split the data
    - train & validation subset(s) for training
    - test subset for final evaluation
  - Standardize the data = ML algorithms behave better with numbers at the same dimension, and usually close to unity
    - min-max = *(X-min(X))/(max(X)-min(X))*
    - standard scaler = *(X-mean(X))/std(X)*
- We need to choose a model
- **We fit the model to the data and predict things!**

The steps involved

- Prediction of the density of Na_2O-K_2O-Al_2O_3-SiO_2 glasses
  - *Data importation*
  - *Data preparation*
  - *Model selection*
  - *Fit data and get predictions!*



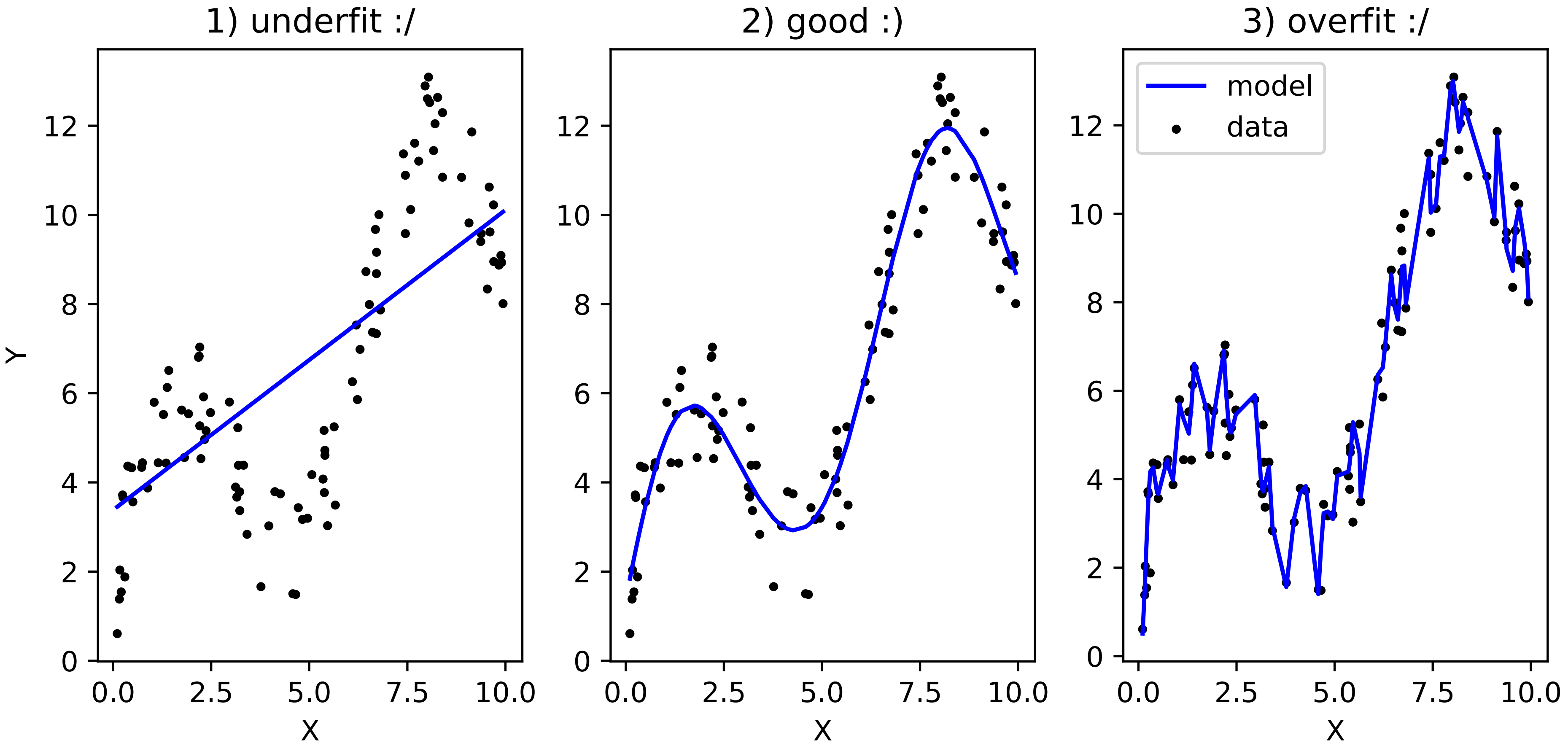
Support Vector Machines

Fit well!

RMSE train: 0.01
RMSE test: 0.01

No ***overfitting***

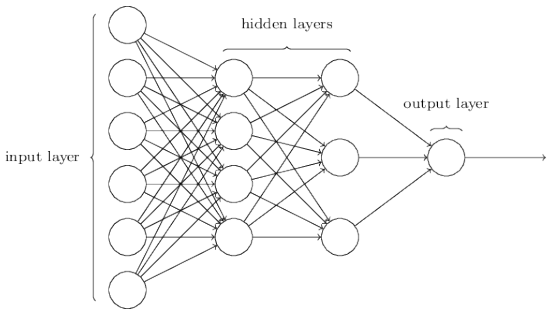
Overfitting

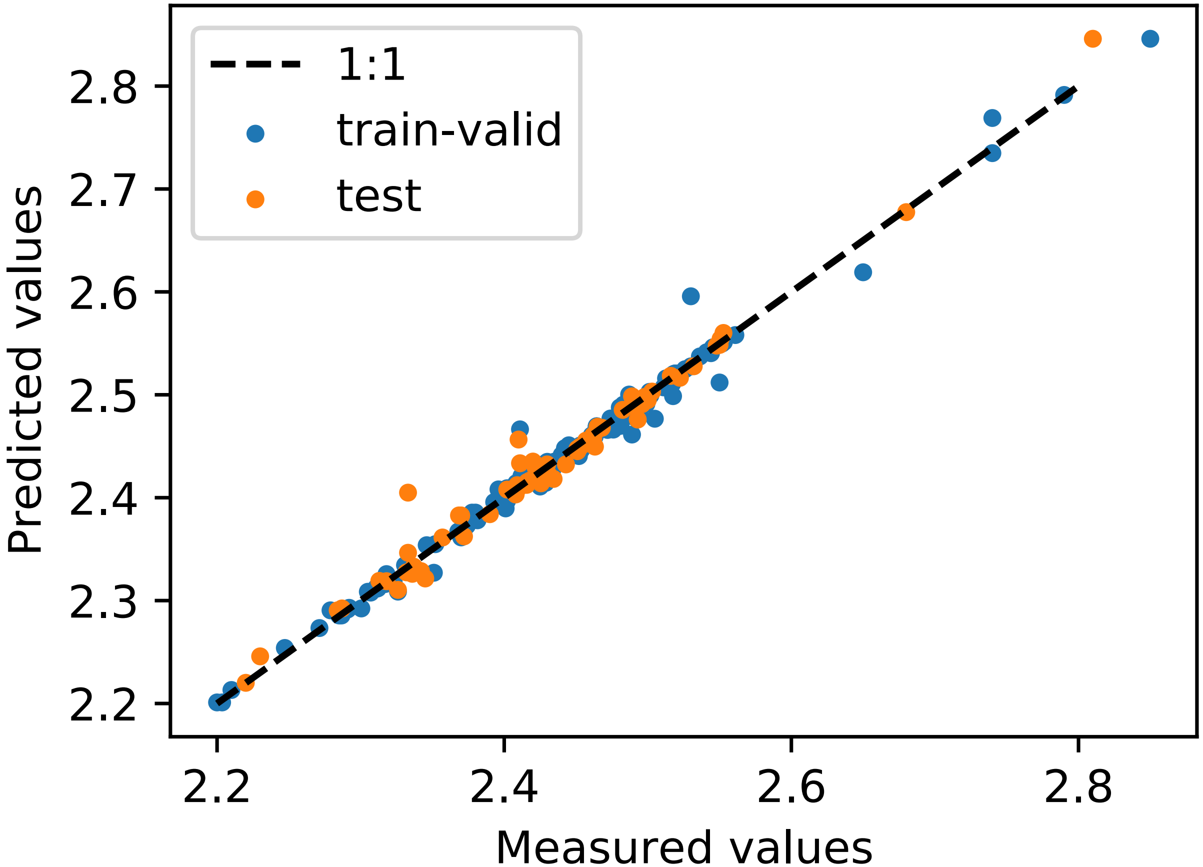


- Algorithm and mitigation methods
- Dataset size… this is why ML develops with the « big data » era!

The steps involved

- Prediction of the density of Na_2O-K_2O-Al_2O_3-SiO_2 glasses
  - *Data importation*
  - *Data preparation*
  - *Model selection*
  - *Fit data and get predictions!*





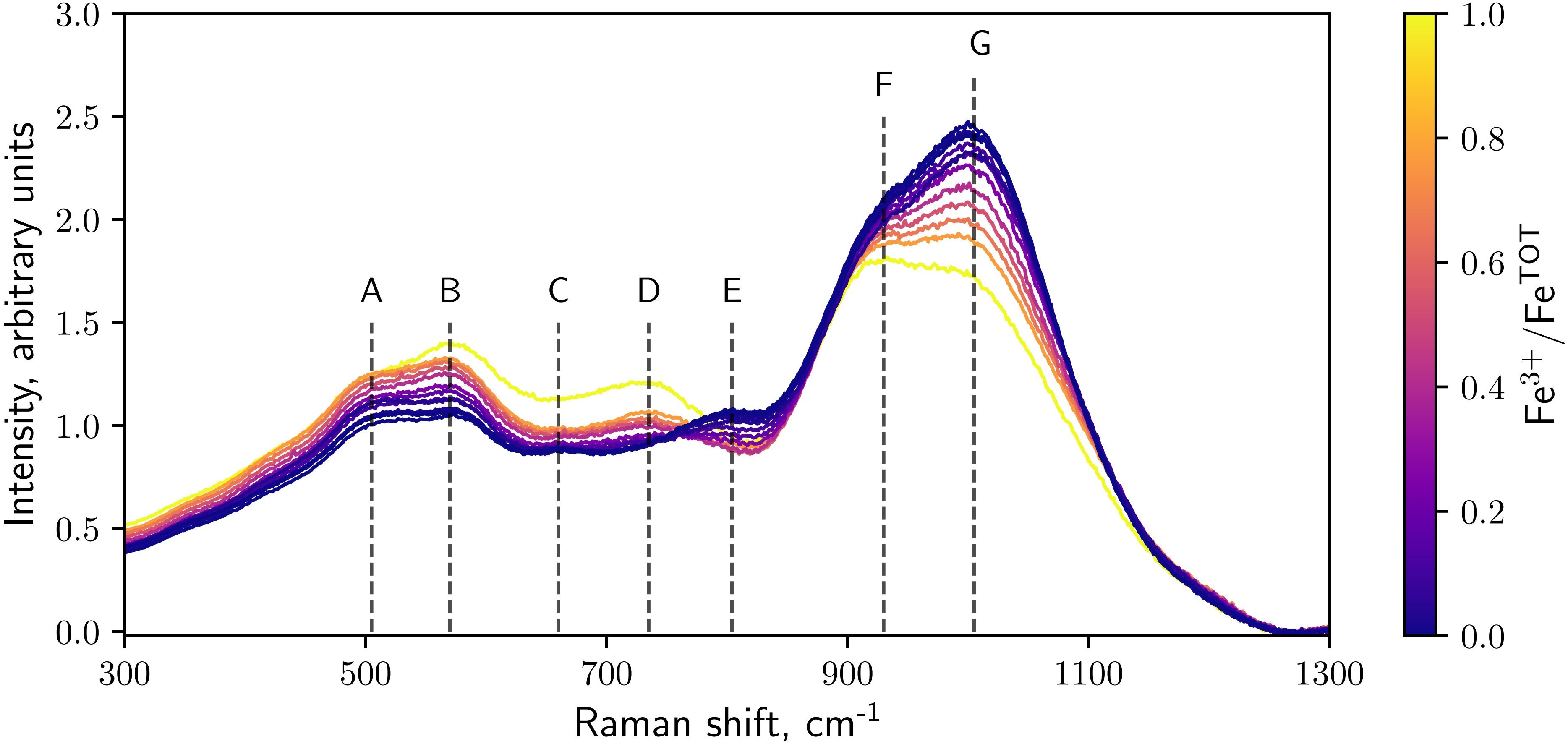
Multilayer perceptrons

Overfitting mitigated via
> hyperparameter tuning
> training strategies

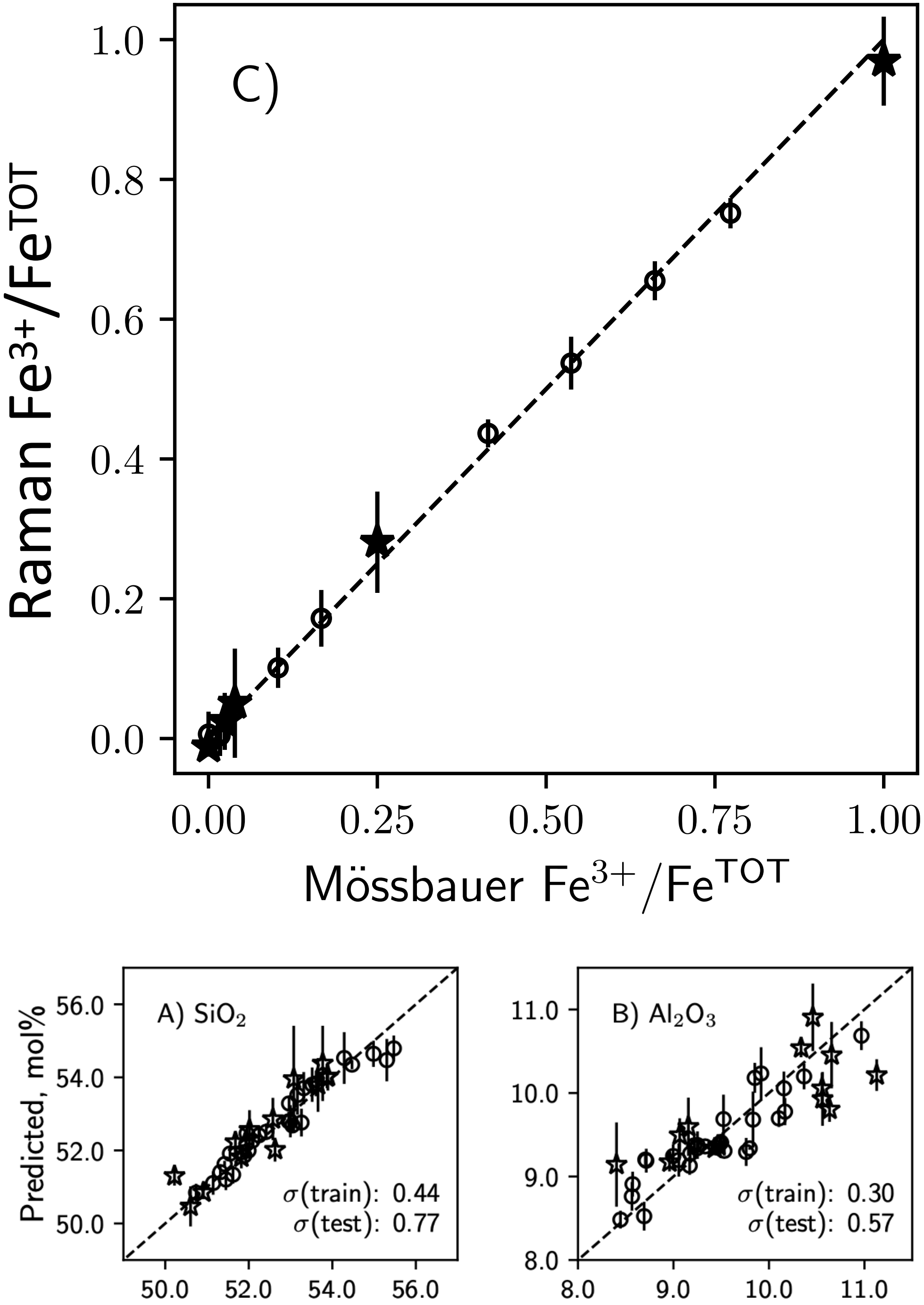
> RMSE train: 0.01
> RMSE test: 0.01

More about Neural Nets for Analyzing data

example: Raman spectra of Fe-bearing CMAS glasses



Le Losq et al., 2019,
Am. Min. 104:1032



Neural Nets for Analyzing data

Artificial neural networks + Raman spectroscopy
=
redox and chemical composition of glasses

Le Losq et al., 2019, Am. Min. 104:1032

Making smarter models

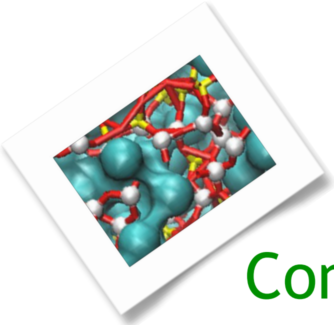
- **Greybox models**
  - a.k.a. physics-based ML models
  - a.k.a. physics and chemistry informed ML models
  - a.k.a. knowledge based ML models
  - a.k.a. semiphysical ML models
  - … !

ML + physico-chemical equations
= Greybox models

Making smarter models

- **Greybox models**
  - a.k.a. physics-based ML models
  - a.k.a. physics and chemistry informed ML models
  - a.k.a. knowledge based ML models
  - a.k.a. semiphysical ML models
  - … !

ML + physico-chemical equations
= Greybox models

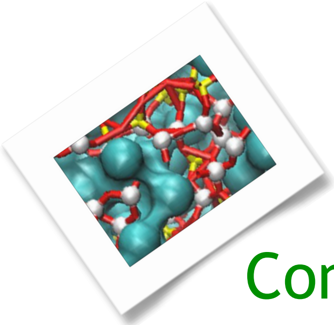


Blackbox:
Composition, T —> Neural Network —> Viscosity

Making smarter models

- **Greybox models**
  - a.k.a. physics-based ML models
  - a.k.a. physics and chemistry informed ML models
  - a.k.a. knowledge based ML models
  - a.k.a. semiphysical ML models
  - … !

ML + physico-chemical equations
= Greybox models



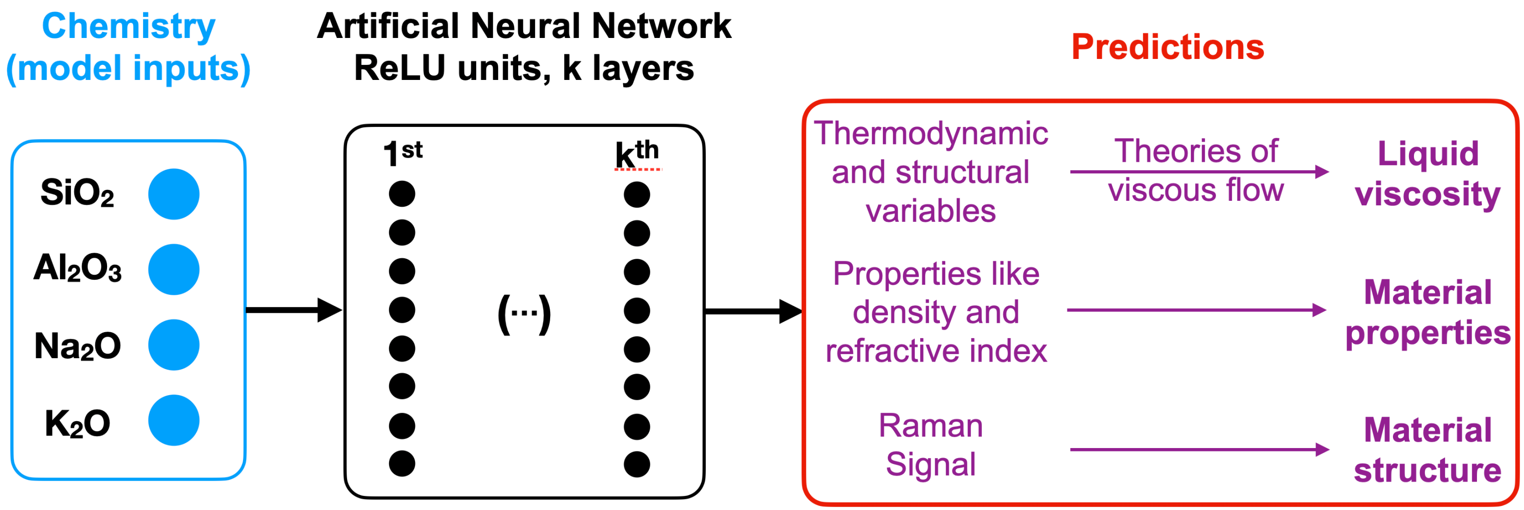
Blackbox:
Composition, T —> Neural Network —> Viscosity

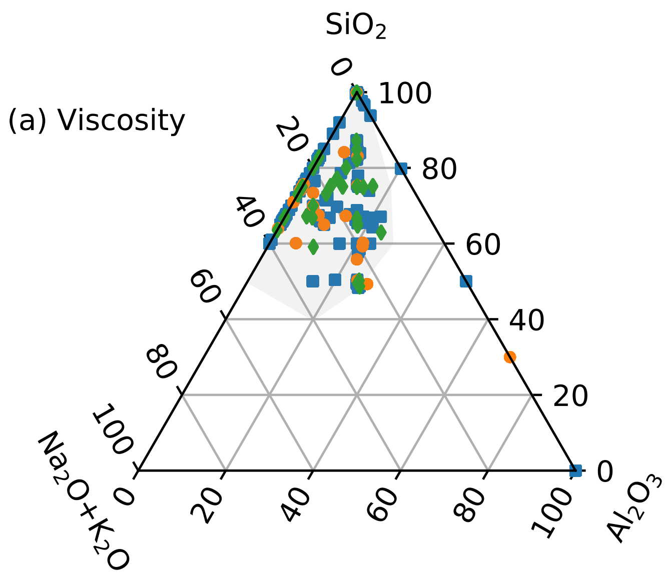
Greybox:
Composition —> Neural Network —> Adam-Gibbs + T —> Viscosity

Making smarter models

For glass: example of *i-Melt*

github.com/charlesll/i-melt/





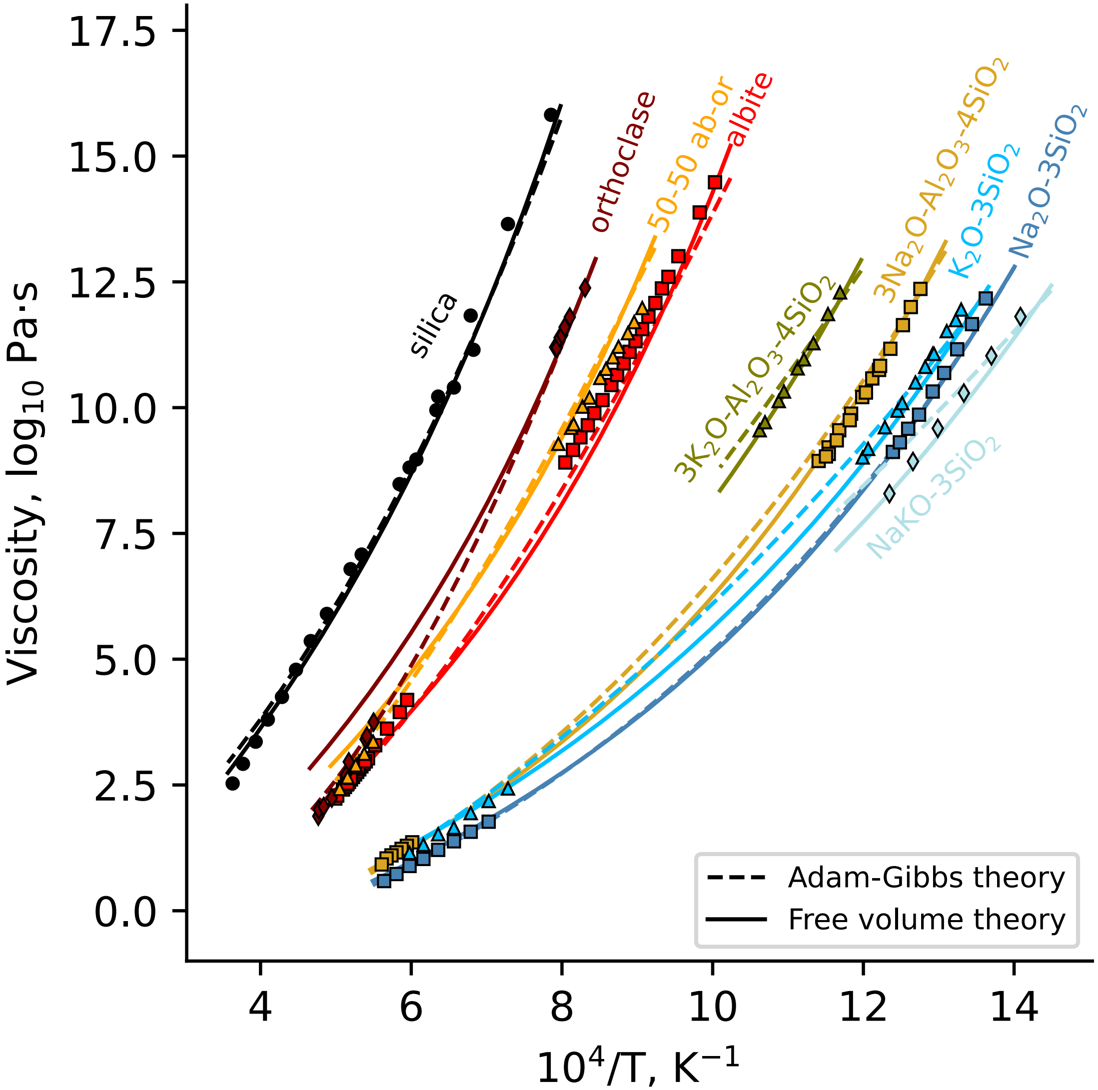
Demonstration on the Na_2O-K_2O-Al_2O_3-SiO_2 system

Le Losq et al., 2021, GCA 314:27

Making smarter models

For glass: example of *i-Melt*

github.com/charlesll/i-melt/



Na_2O-K_2O-Al_2O_3-SiO_2 system

Viscosity ± 0.4 log unit

Density ± 0.02 g cm^{-3}

Refractive index ± 0.006

T_g ± 19 K

$S^{conf}(T_g)$ ± 0.9 J mol^{-1} K^{-1}

Raman spectra ± ~20%

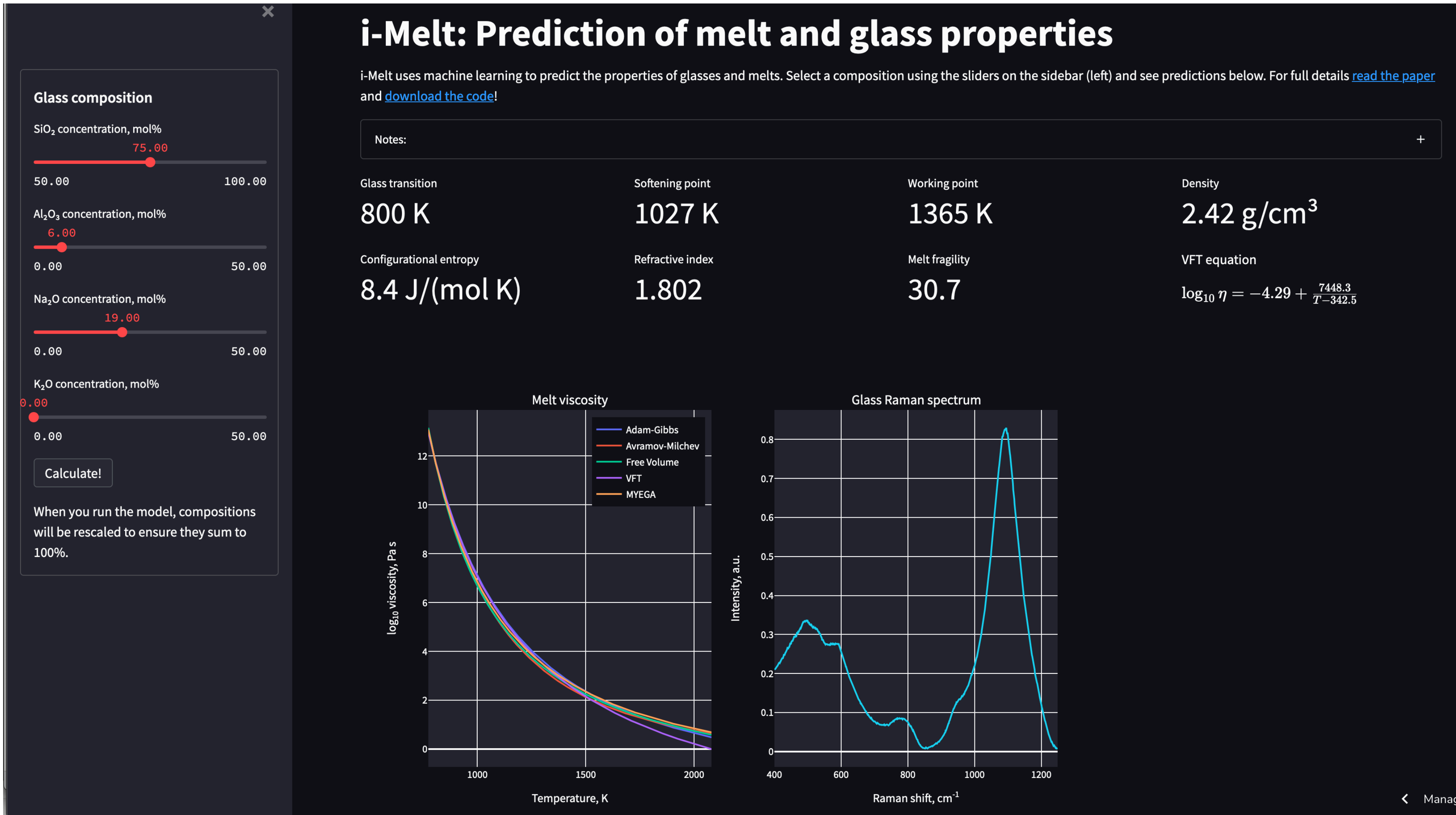
Parameters of Adam-Gibbs, Free Volume, MYEGA, VFT, Avramov-Milchev equations…

Le Losq et al., 2021, GCA 314:27

Making smarter models

For glass: example of *i-Melt*

github.com/charlesll/i-melt/

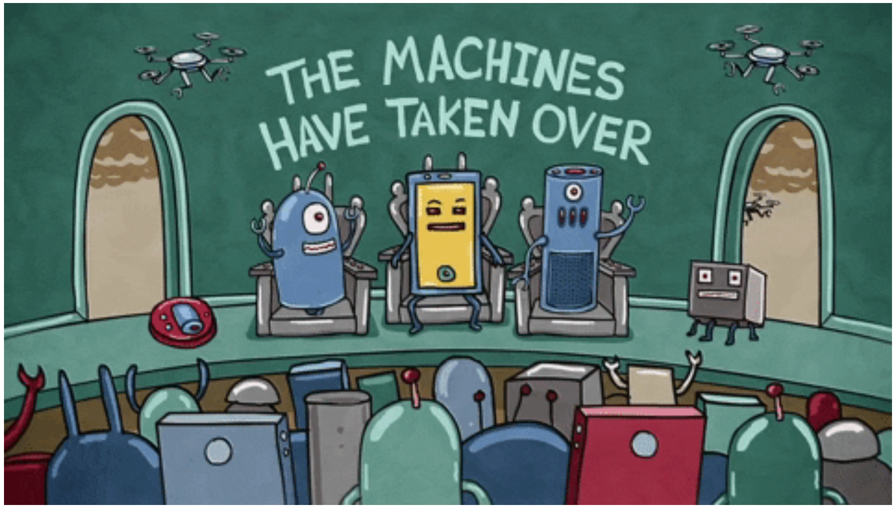


Le Losq et al., 2021, GCA 314:27

Conclusion

Theory - Models - Observation - new: machine learning

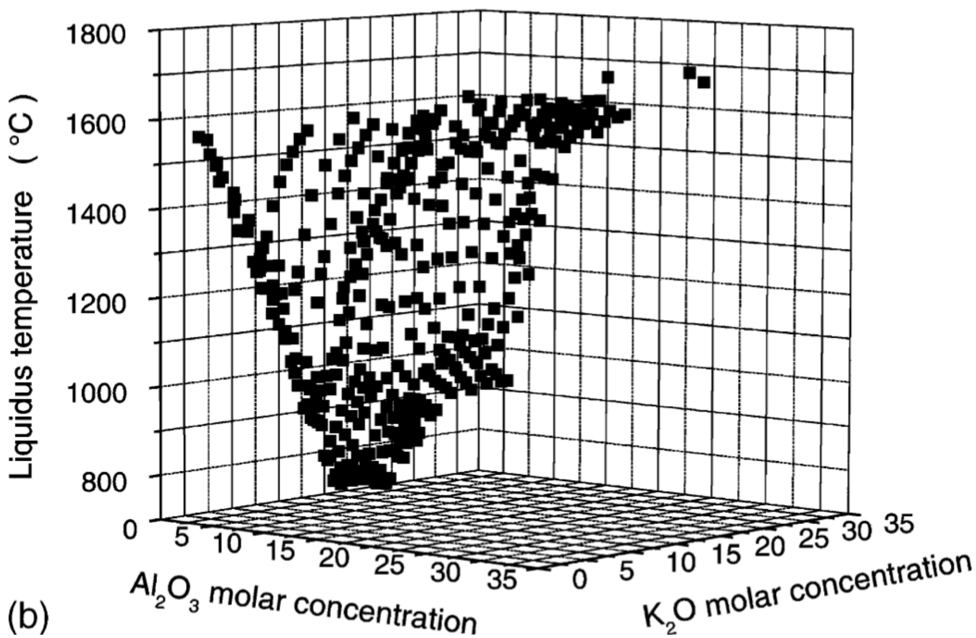
- analyze data and generate predictive models of properties
- explore datasets
- select/design new compositions
- detect drifts & anomalies
- …

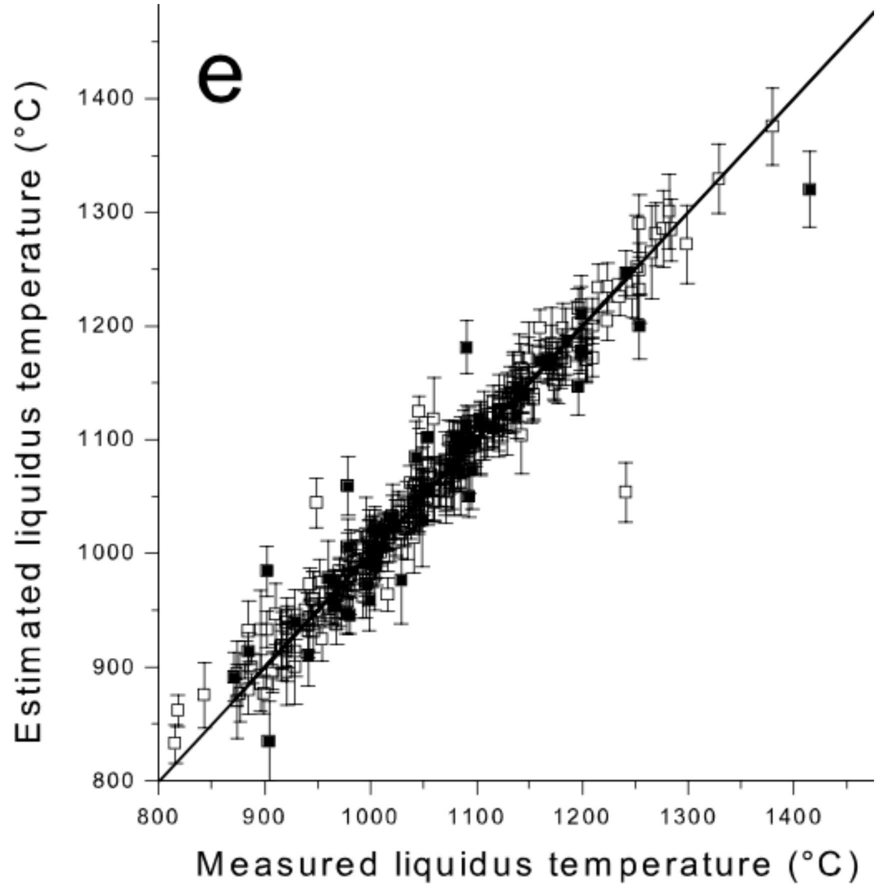


The challenges

- datasets (generation, quality…)
- algorithms and uncertainties
- interpretation
- automated systems for industry
- balance between blackbox & physics

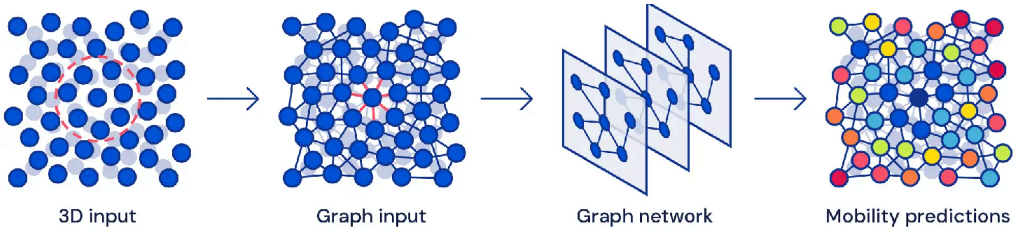
- **Not « new »**
- **Not « artificial intelligence »**

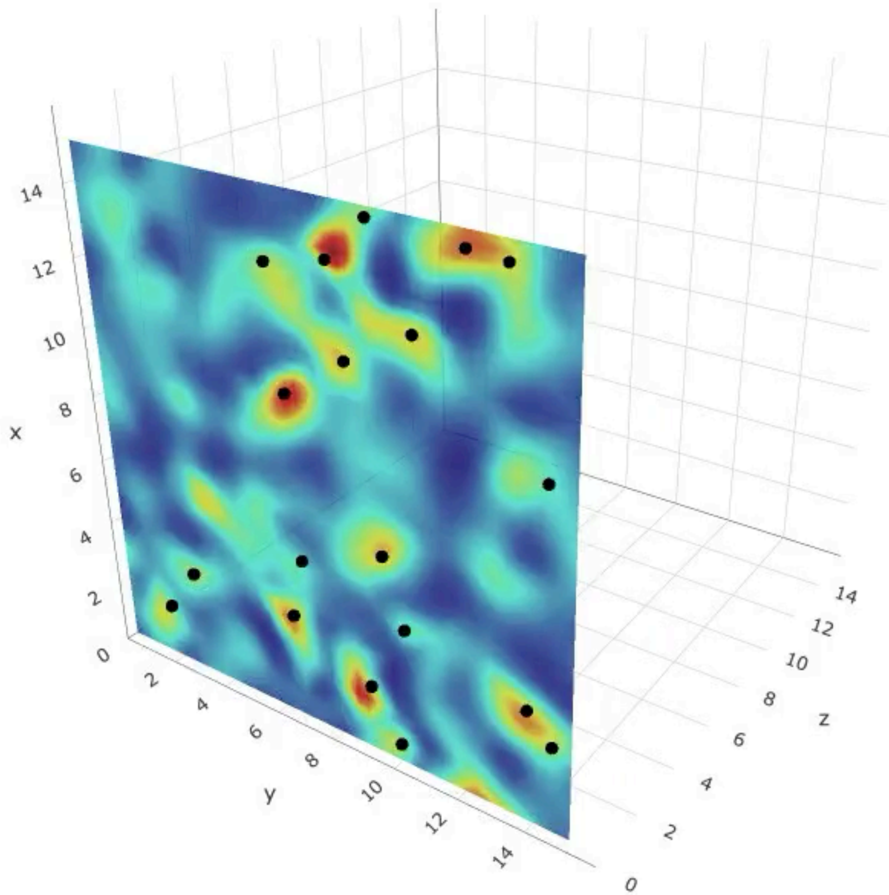




Example: prediction of liquidus temperatures of K_2O-Al_2O_3-SiO_2 using Neural Networks by Dreyfus et Dreyfus, 2003

Examples of applications





Bapst et al. 2019

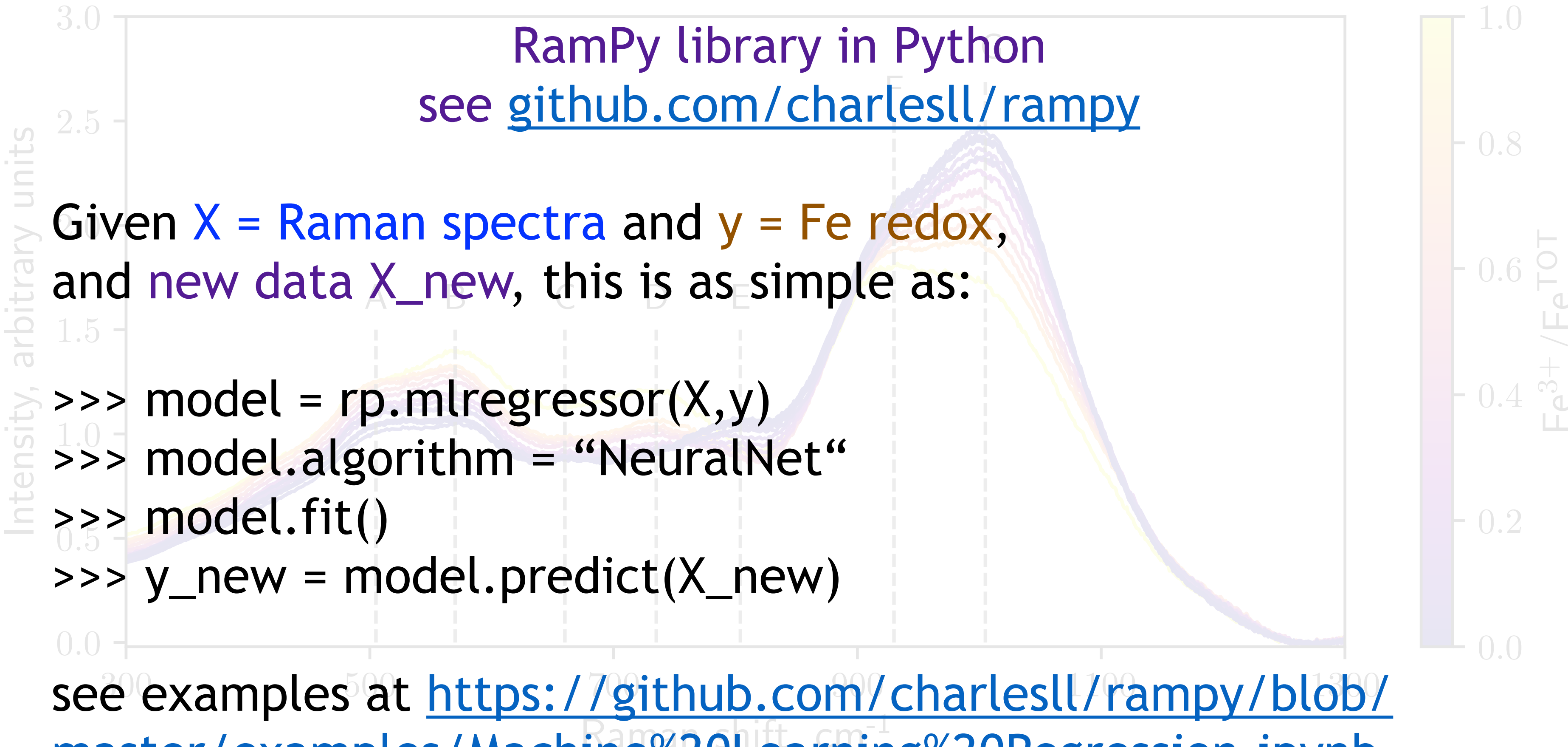
DeepMind/Google

Prediction of atomic movements via graph neural networks

Better understand properties & glass transition

More about Neural Nets for Analyzing data

example: Raman spectra of Fe-bearing CMAS glasses



RamPy library in Python
see github.com/charlesll/rampy

Given X = Raman spectra and y = Fe redox,
and new data X_new, this is as simple as:

```
>>> model = rp.mlregressor(X,y)
>>> model.algorithm = “NeuralNet“
>>> model.fit()
>>> y_new = model.predict(X_new)
```

see examples at https://github.com/charlesll/rampy/blob/master/examples/Machine%20Learning%20Regression.ipynb

Classification of boninite magmas

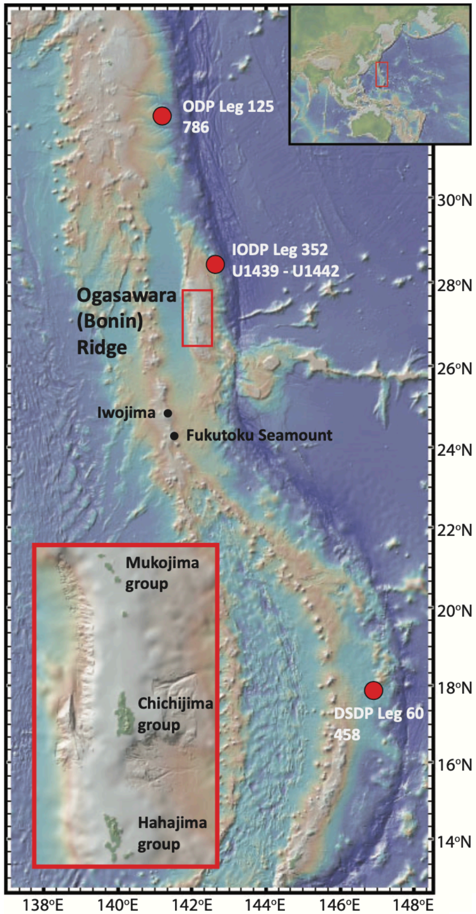
> MgO-rich & TiO_2 poor magmas, formed by partial melting of clinopyroxene-bearing harzburgite sources

> Different sources = different trends, difficult to distinguish in MgO-SiO_2

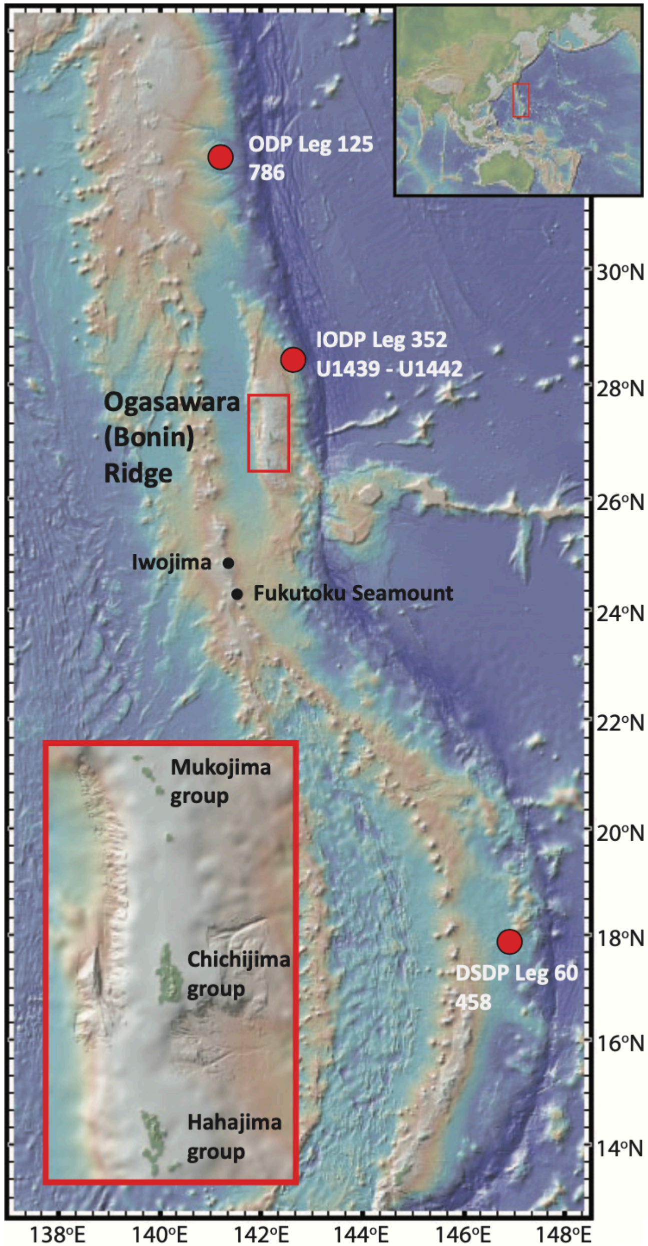
> For Ogasawara Ridge (Kanayama et al. 2012):

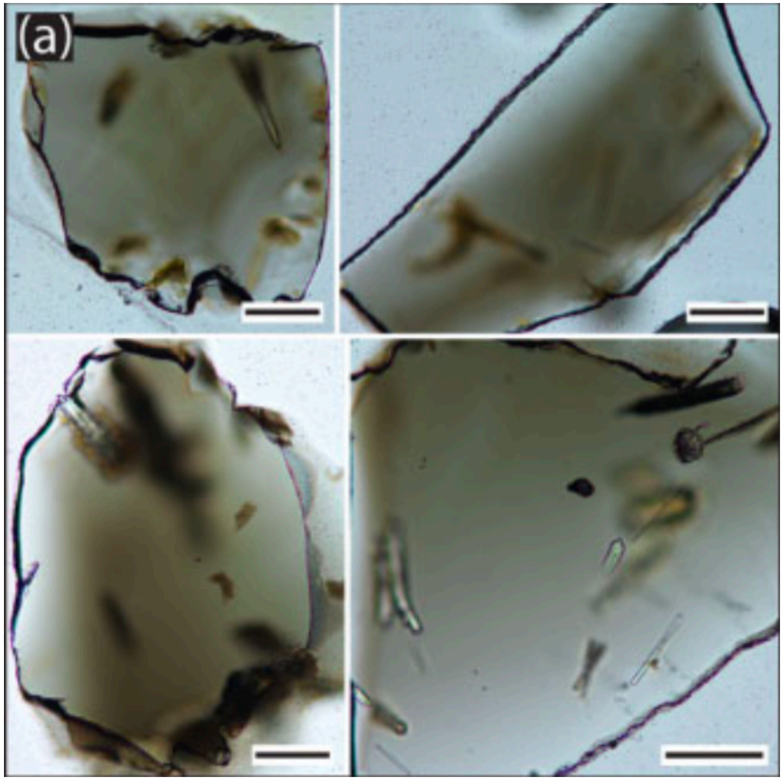
- HSB: High Silica Boninite
- LSB: Low Silica Boninite

LSB = less depleted mantle source, lower T



Classification of boninite magmas

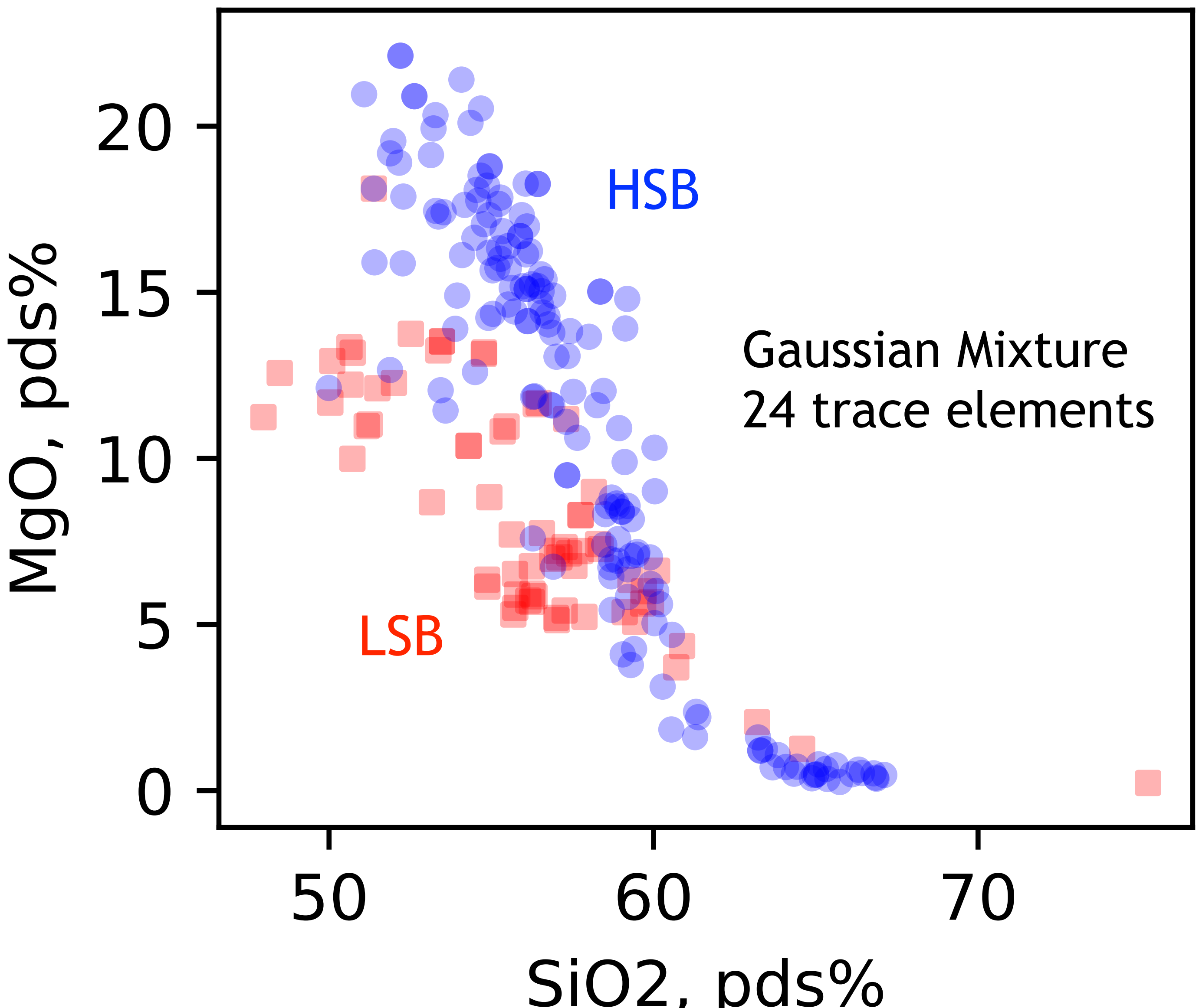




bars = 100 microns

glass shards analysis (major elements + trace analysis)

Classification of boninite magmas



Classification of boninite magmas

> Different sources = different trends, difficult to distinguish in $MgO-SiO_2$

> Easy for a Gaussian mixture classification algorithm, unsupervised, trained on REE LA-ICP-MS data

Valetich, Le Losq, Arculus, Umino, Mavrogenes, Journal of Petrology 2021
github.com/charlesll/boni-and-class