Improving Operational Geomagnetic Index Forecasting

Laurence Billingham [laurence@bgs.ac.uk], Gemma Kelly
British Geological Survey, West Mains Road, Edinburgh, UK

1. Introduction

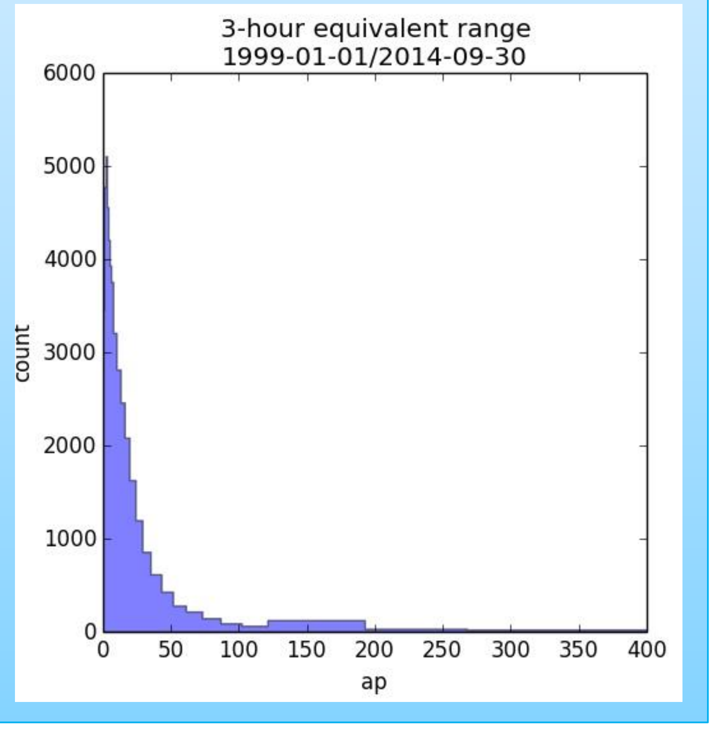
The interest in space weather has never been greater, with society becoming ever more reliant upon technology and infrastructure which are potentially at risk. Geomagnetic storms are potentially damaging to power-grids, communication systems and oil and gas operations.

Geomagnetic indices

- Capture magnetic storm severity by summarising lots of data
- have become ubiquitous parameterisations of storm-time magnetic conditions
- required as inputs by a variety of models

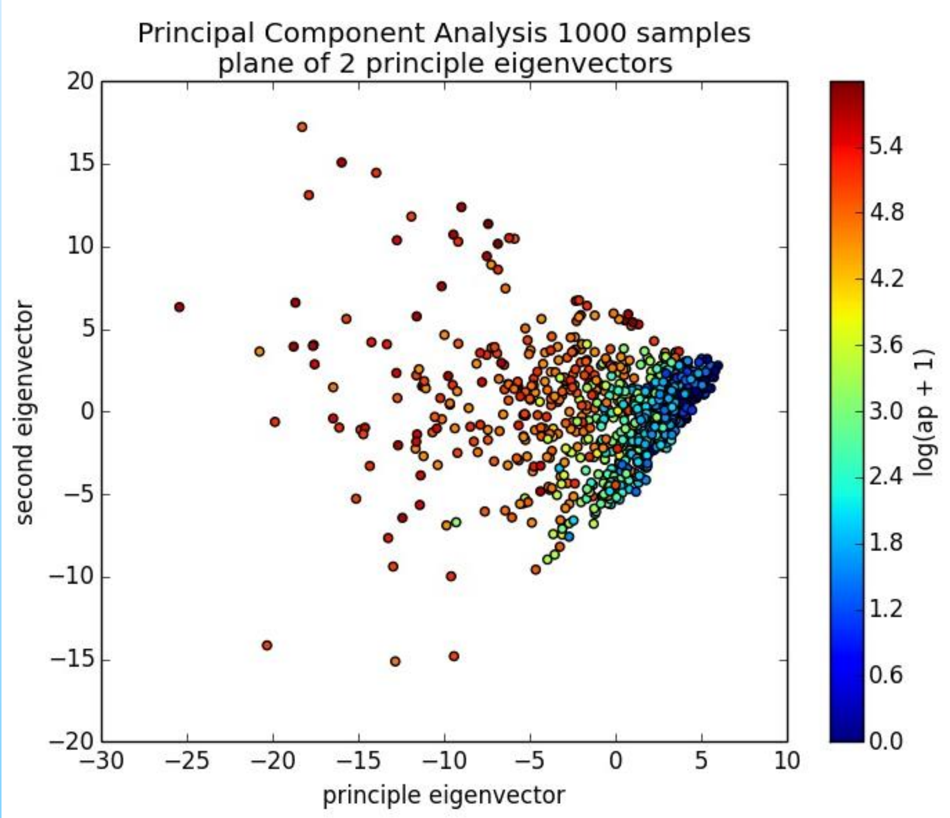
a_p index

- captures amplitude of the disturbance in horizontal part of the field (see e.g. [1] for more detail)
- tracks disturbances within a 3-hour interval
- indicates the global level of disturbance



2. Data

- Samples times over ~15 years of geomagnetic and solar wind data
- Storms rare but important
  - **Balance dataset** otherwise storms look like noise
- **Features** selected like
  - $\min(\vec{B}_{IMF\ z})_{last\ hour}$
  - $(a_p)_{2\ intervals\ ago}$
  - $\mathrm{mean}(v_{SW})_{3\ hours\ ago}$
- **Split**: training set, validation set, test set
- Training set **scaled**
  - $x_{sample} \rightarrow (x - \bar{x})/s$
  - Same scaling applied to other sets
- Some algorithms require $x_i \not\propto x_j$
  - use **Principal Component Analysis** to decompose



3. Techniques

$$\vec{y} = \begin{bmatrix} a_{p\ 1} \\ a_{p\ 1} \\ \vdots \end{bmatrix} \qquad \mathbf{X} = \begin{array}{c} \xrightarrow{n_{features}} \\ \begin{bmatrix} x_{11} & x_{12} & \cdots \\ x_{21} & x_{22} & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix} \end{array} \downarrow n_{samples}$$

Machine Learning

- A branch of statistics
- We use **regression** algorithms here
- Data laid out as for matrix inversion (little like finding best fit line with 2D data)
- Many algorithms (see [2] for an excellent introduction), some are like linear regression e.g.

$$\boxed{\min_{\mathbf{w}}\left(\boxed{||\mathbf{X}\mathbf{w} - \vec{y}||_2^2} + \boxed{\alpha\rho||\mathbf{w}||_1} + \boxed{\frac{\alpha(1-\rho)}{2}||\mathbf{w}||_2^2}\right)}$$

Linear Regression LR + = Lasso LR + = Ridge

LR + Lasso + Ridge = ElasticNet

- Workflow:
  - **Training**: get coefficients $\mathbf{w}$ from $\mathbf{X}\mathbf{w} \stackrel{?}{=} \vec{y}_{new}$
  - **Tune** model parameters against validation set
  - **Test and score** model with test set $\mathbf{X}^{-1}\vec{y}_{known} = \mathbf{w}$
  - **Predict** new a_p from unseen data

ARIMA

- Auto-regressive moving average
- A linear regression over a windowed average of a_p
- Only input is a_p timeline
- Currently operational: used here as a baseline quality comparison

4.Results

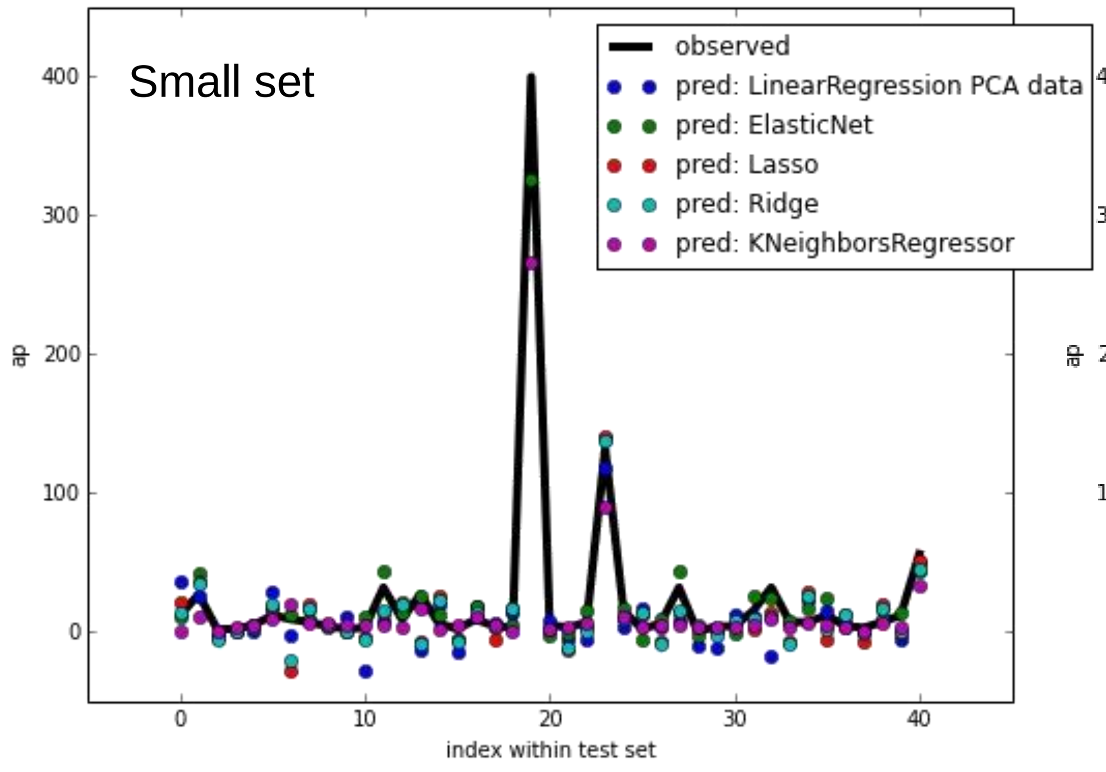
- Initial dataset with 205 samples (small set)
  - Some models much better at identifying storms than others
  - Large range in rms values and percentage of predictions which are close to the true value
- We then increased the total dataset size to 1000 samples (large set) and tested the best performing models
  - Again range of rms values
  - All the machine learning models out perform the ARIMA model in terms of rms, HitRate and skill (HSS)
- Positive results: worth pursuing for production system

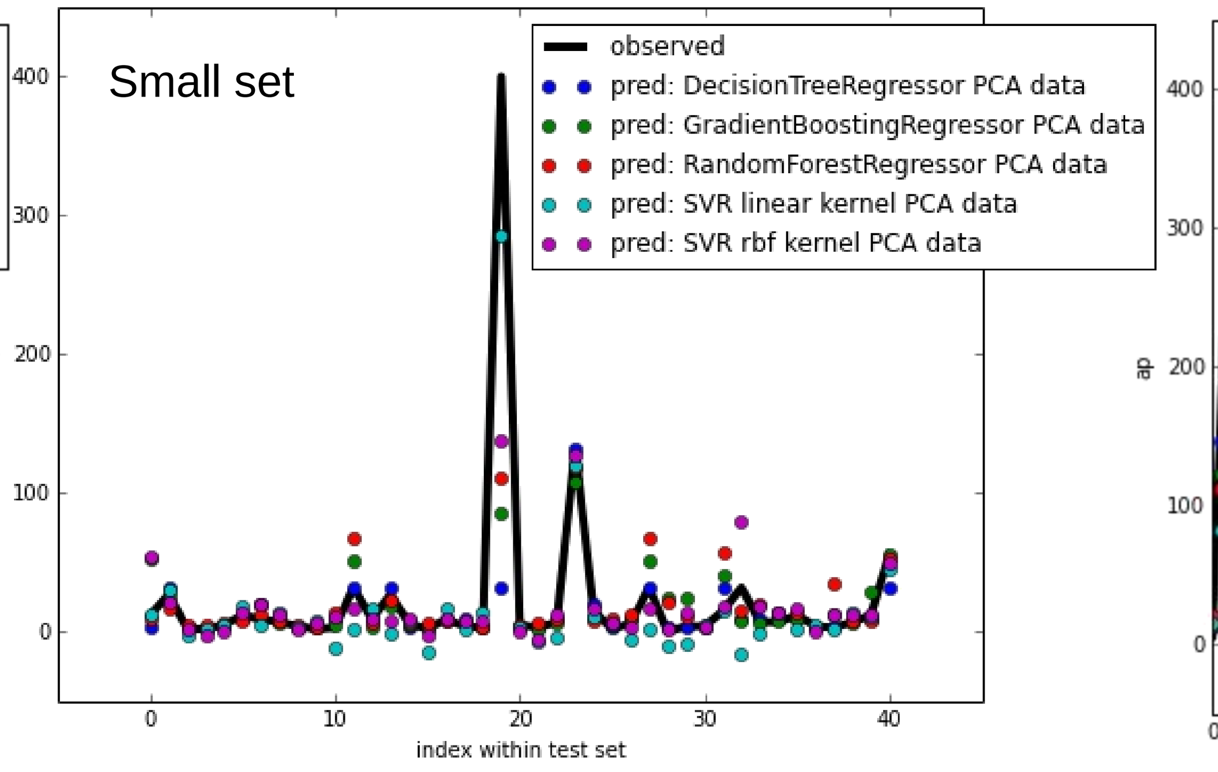
	rms	% within ± 5	% within ± 10	HitRate	HSS	FAR
Ridge	12.37	29.27	60.98	1.00	1.00	0.00
Lasso	17.63	26.83	53.66	1.00	1.00	0.00
ElasticNet	15.78	56.10	82.93	1.00	0.63	0.50
LinearRegression	14.88	31.71	51.22	1.00	1.00	0.00
DecisionTreeRegressor	17.96	73.17	87.80	0.33	0.48	0.00
GradientBoostingRegressor	36.29	68.29	73.17	1.00	0.55	0.57
RandomForestRegressor	25.42	60.98	73.17	1.00	0.63	0.50
KNeighborsRegressor	21.81	65.85	75.61	0.67	0.79	0.00
SVR_linear_kernel	23.93	41.46	63.41	1.00	1.00	0.00
SVR_rbf_kernel	43.65	48.78	78.05	1.00	0.72	0.40
1000 samples						
DecisionTreeRegressor	41.40	16.42	52.24	0.89	0.84	0.11
RandomForestRegressor	34.26	56.22	69.65	0.97	0.89	0.12
ElasticNet	36.48	34.33	56.72	0.95	0.91	0.08
SVR	38.18	35.82	63.68	0.90	0.88	0.07
LinearRegression	37.26	34.83	54.73	0.95	0.91	0.08
ARIMA	49.04	59.79	70.10	0.82	0.84	0.04

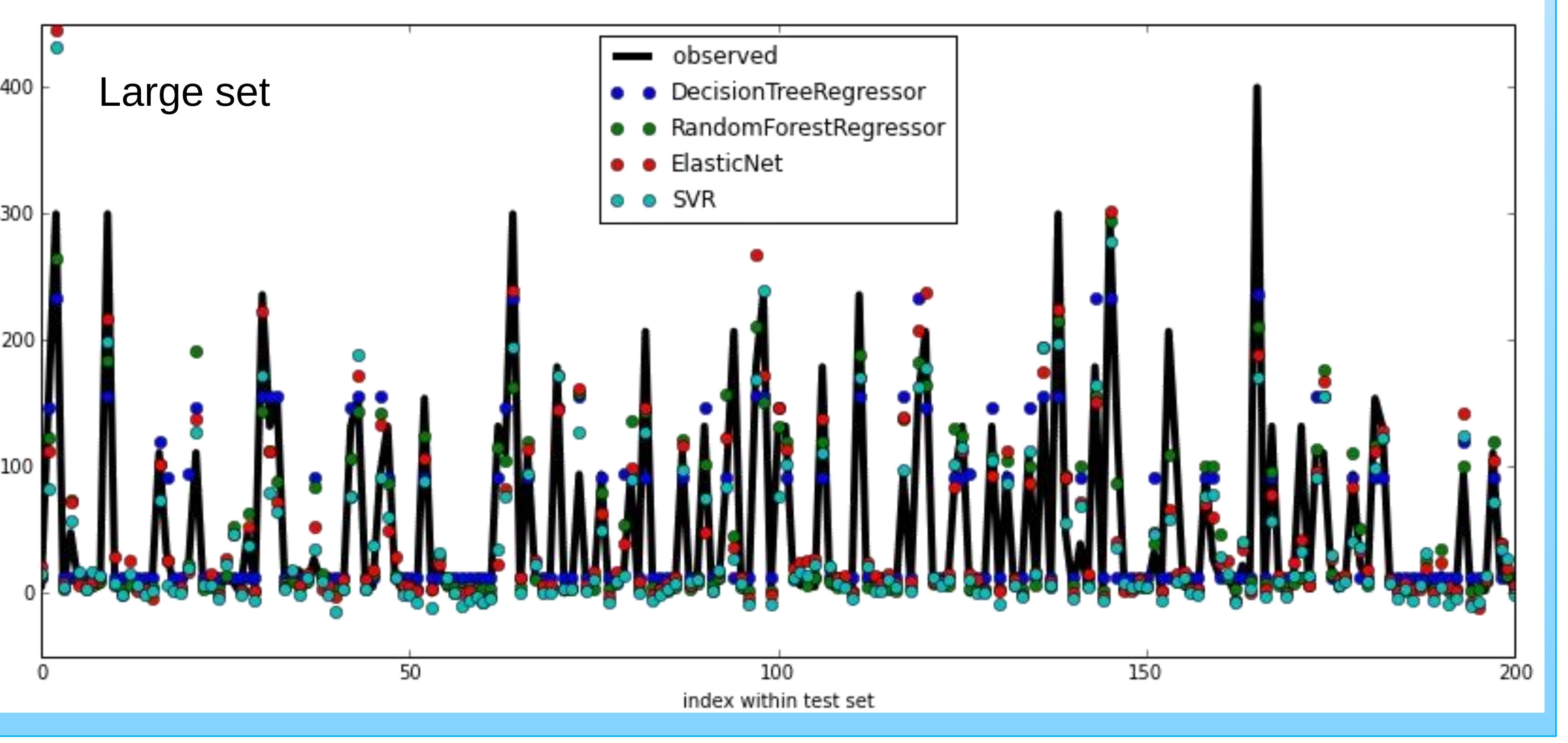
Metrics:

- **rms:** root-mean square error
- **% within ±N:** Percentage of predicted values within ±N of the observed value
- **HitRate:** how well do we predict the storms?
  - 1 = predicted every single storm
  - 0 = missed every storm
- **HSS:** Heidke skill score measures fractional improvement of the forecast over forecast by random chance
  - HSS = 2 (ad – bc) / [(a+ c)(c + d) + (a + b)(b + d)]
  - 1 = highly skilled
  - 0 = no skill
  - <0 = worse than random chance
- **FAR:** False alarm rate of storm prediction
  - 0 = no false alarms
  - 1 = all false alarms

Event Forecast	Storm Observed		
	Yes	**No**	**Forc Σ**
Yes	a	b	a + b
No	c	d	c + d
Obs Σ	a + c	b + d	a+b+c+d = n

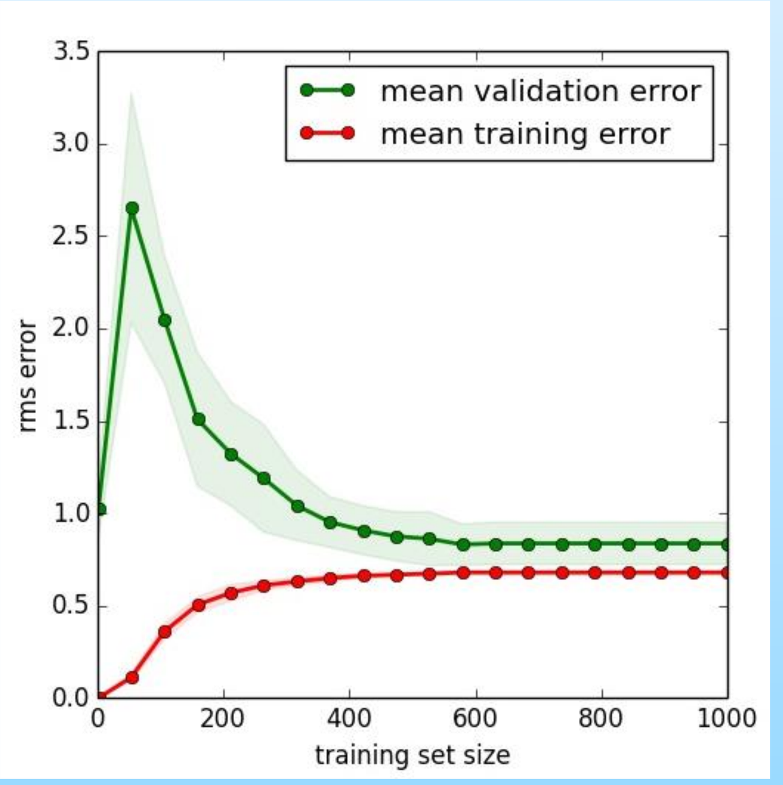






5.Summary and Future Work

- Scoping study results positive
  - value in predictions
  - proceed to operational system
- Here we only predict 1 a_p interval into future
  - Some models easily configures to predict multiple intervals
  - Others need new train, validate, test cycles
- Classification not regression
  - e.g. G1, ..., G5
  - More useful aid to human forecaster
  - Potentially easier computation
  - Up-weight storm categories: balance dataset
- More features per sample
  - Models converge with few training samples (see fig): models powerful enough
  - Data mine human forecasts, coronagraph data ...
  - Science potential in 'white-box' models: which features give useful info?



References
[1] McPherron, Magnetospheric Dynamics, in *Introduction to Space Physics*, edited by Kivelson, Russell, pp. 400-458, Cambridge University Press, 1995.
[2] Hastie *et al.*, *The Elements of Statistical Learning Data Mining, Inference, and Prediction*, Springer 2009(II)

This work is powered by Python-Scikit-learn
Pedregosa *et al.*, *Scikit-learn: Machine Learning in Python*, JMLR 12, pp. 2825-2830, 2011.