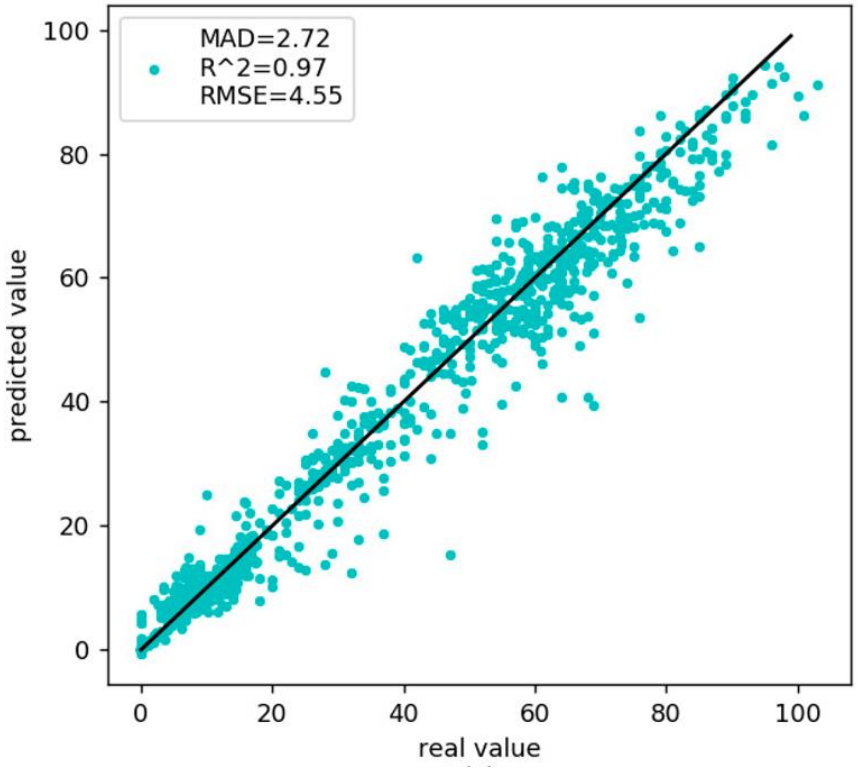
Regression

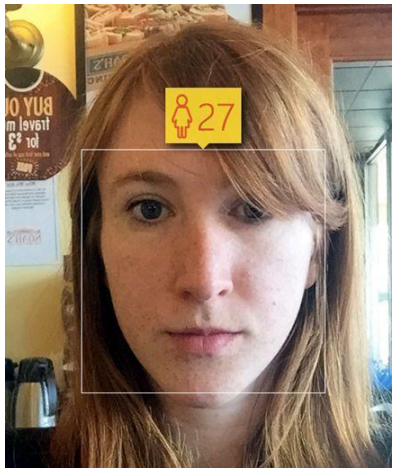
Dr. Fayyaz Minhas

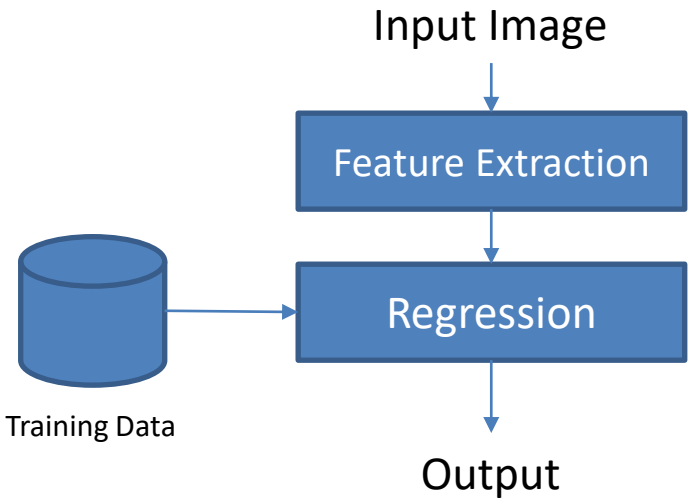
Department of Computer Science
University of Warwick
https://warwick.ac.uk/fac/sci/dcs/teaching/material/cs909/

Problems Other than Classification

- Regression
  - Output is a continuous variable
  - Predict age from an image

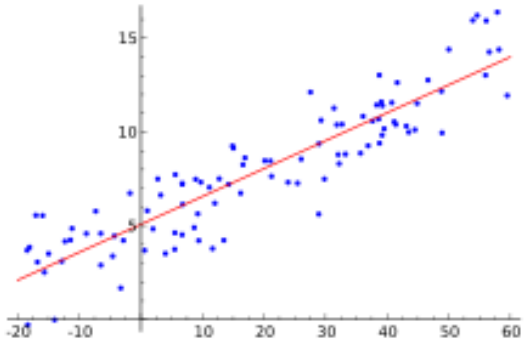






Regression

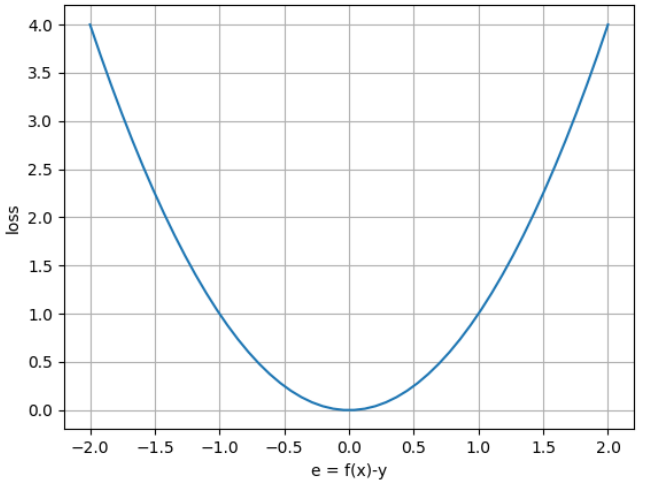
- Estimate the relationship among variables
  - Dependent variables
  - Independent variables
- Used in
  - prediction and forecasting
  - estimating effects of variables on each other
- Mathematical formulation
  - **Model:** $y = f(\boldsymbol{x}; \boldsymbol{w}) + \boldsymbol{\epsilon}$
    - **For example:** $y = wx + b + \boldsymbol{\epsilon}$
  - The Objective is to estimate the parameters $\boldsymbol{w}$ that produce outputs that are close to the given targets



x	y
1	3.5
2	4.75
3	7.05
4	9.5
…	…

SRM: Ordinary Lease Squares Regression (OLSR)

- Representation
  - $f(\boldsymbol{x}) = \boldsymbol{w^T x} + b$
- Evaluation
  - Regularization: None
  - Error: Square Loss
  - $l(f(x), y) = (\boldsymbol{f(x)} - \boldsymbol{y})^2$
- Optimization Problem



$$\min_{w} \sum_{i=1}^{N} (\boldsymbol{w^T x_i} - y_i)^2 = \|\boldsymbol{Xw} - \boldsymbol{y}\|^2 = (\boldsymbol{Xw} - \boldsymbol{y})^T (\boldsymbol{Xw} - \boldsymbol{y})$$

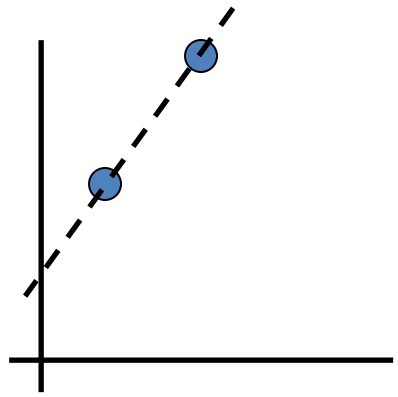
$$\boldsymbol{X}_{(N\times d)}, \boldsymbol{y}_{(N\times 1)} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$$

Taking derivative wrt w and setting to zero gives:

$$\boldsymbol{w} = \left(\boldsymbol{X^T X}\right)^{-1} \boldsymbol{X^T y}$$

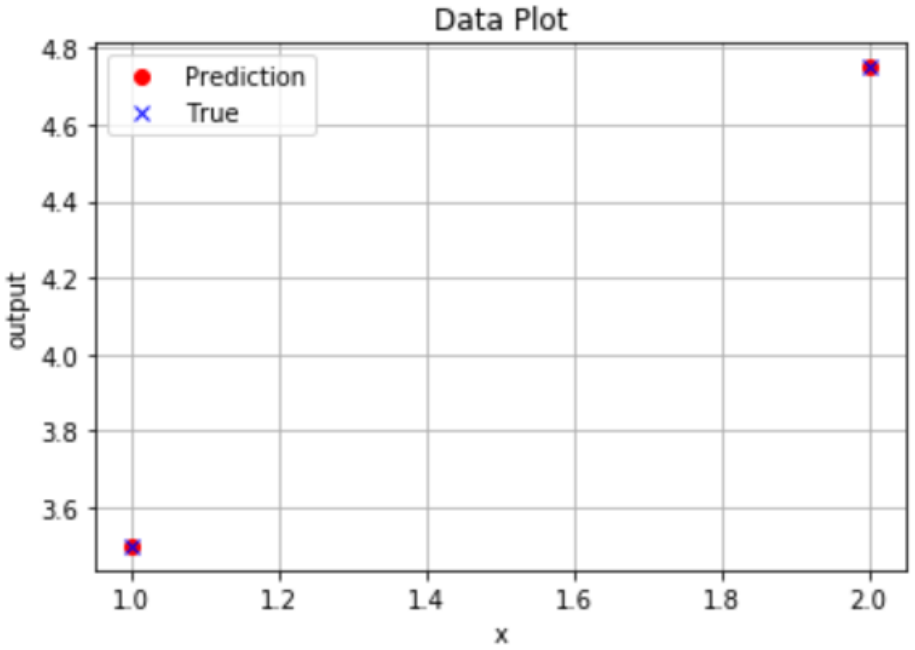
Linear Regression: Simple(st) Example

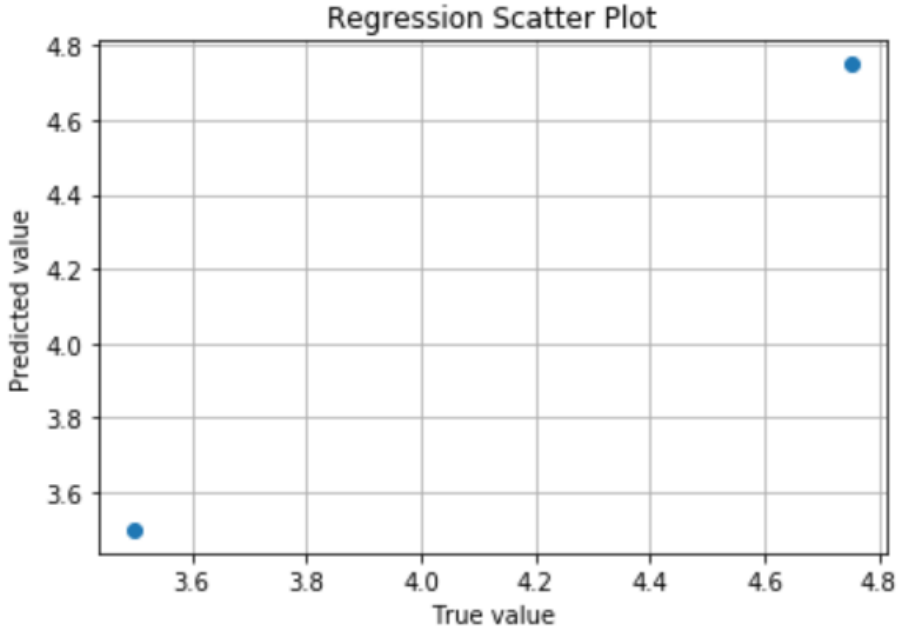
- Example
  - $\boldsymbol{X} = \begin{bmatrix} 1 & 1 \\ 2 & 1 \end{bmatrix}, \boldsymbol{y} = \begin{bmatrix} 3.5 \\ 4.75 \end{bmatrix}$
  - Thus: $\boldsymbol{w} = \left(\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{X}\right)^{-1}\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{y} = \begin{bmatrix} 1.25 \\ 2.25 \end{bmatrix}$
  - Now
    - $\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{x}^{(1)} = \begin{bmatrix} 1.25 \\ 2.25 \end{bmatrix}^{T} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = 3.5$
    - $\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{x}^{(2)} = \begin{bmatrix} 1.25 \\ 2.25 \end{bmatrix}^{T} \begin{bmatrix} 2 \\ 1 \end{bmatrix} = 4.75$



x	y
1	3.5
2	4.75

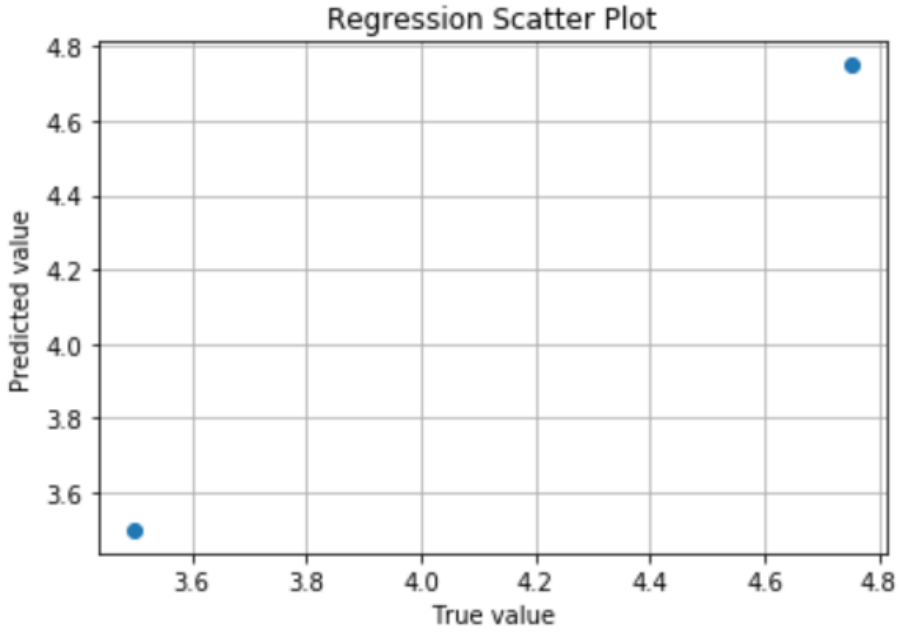
Coding






```
import numpy as np
import matplotlib.pyplot as plt
X0 = np.array([[1],[2]])
y =np.array([3.5,4.75])
X = np.hstack((X0,np.ones((X.shape[0],1)))) #append 1 to each example
w = np.linalg.pinv(X)@y
f = X@w
e = f-y
L = e@e
plt.figure();plt.plot(X0,f,'ro');plt.plot(X0,y,'bx');plt.grid();plt.xlabel('x');plt.ylabel('output');plt.legend(['Prediction','True']);
plt.title('Data Plot')
plt.figure();plt.plot(y,f,'o');plt.grid();plt.xlabel('True value');plt.ylabel('Predicted value');plt.title('Regression Scatter Plot')
```

Using Sk-learn



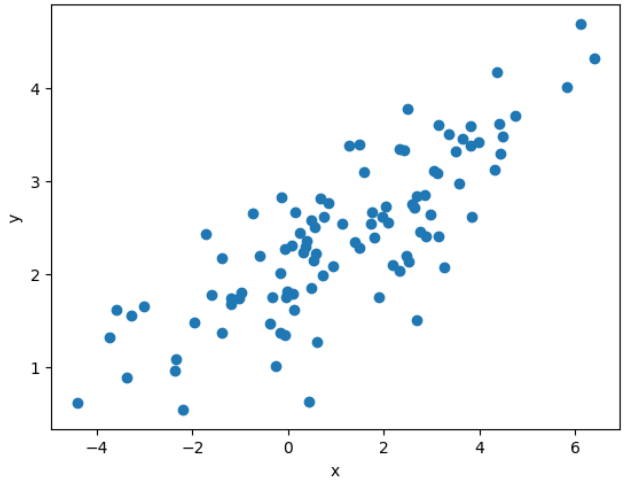

```
from sklearn.linear_model import LinearRegression
regr = LinearRegression(fit_intercept = False).fit(X, y)
f = regr.predict(X)
print('Weights:',regr.coef_)

plt.figure();plt.plot(y,f,'o');plt.grid();plt.xlabel('True value');plt.ylabel('Predicted value');plt.title('Regression Scatter Plot')

# No need to append 1 to feature vector using below
from sklearn.linear_model import LinearRegression
regr = LinearRegression(fit_intercept = True).fit(X0, y)
f = regr.predict(X0)
plt.figure();plt.plot(y,f,'o');plt.grid();plt.xlabel('True value');plt.ylabel('Predicted value');plt.title('Regression Scatter Plot')
```

Statistical Inference by Regression

- Statistical Inference
  - The goal of regression is to identify how a change in one variable changes the expected value of the other variable while keeping all else the same (ceteris paribus)
- Linear regression: E[y|x] = b + w x
  - Statistics libraries write it using the shorthand notation: $y \sim 1 + x$
  - What do the coefficients tell us?
    - **w** tells us if x and y have a (linearly) association
    - We can get "p-values" for the regression coefficient and use this as a hypothesis test to test if the two variables are indeed related
      - In the example in the given code (link below): For each one-unit increase in x, the expected value of y increases by 0.2970 units, holding all other variables constant. This coefficient is also statistically significant, with a very low p-value ($p < 0.05$), suggesting a strong linear relationship between x1 and y.



Synthetic Data Generation

```
# Array of 1000 values with mean = 1.5, stddev = 2.5
X = 2.5 * np.random.randn(1000) + 1.5
# Generate 1000 noise terms
noise = 0.5 * np.random.randn(1000)
# Actual values of Y
y = 2 + 0.3 * X + noise
```

OLS Regression Results

Dep. Variable:	y	R-squared:	0.722
Model:	OLS	Adj. R-squared:	0.721
Method:	Least Squares	F-statistic:	2327.
Date:	Sat, 10 Feb 2024	Prob (F-statistic):	1.57e-251
Time:	02:50:10	Log-Likelihood:	-624.91
No. Observations:	900	AIC:	1254.
Df Residuals:	898	BIC:	1263.
Df Model:	1		
Covariance Type:	nonrobust		

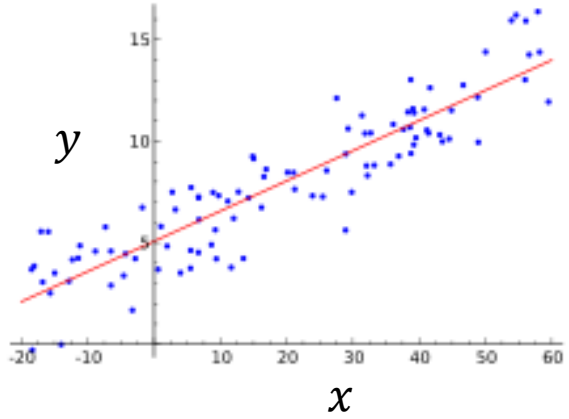
	coef	std err	t	P>|t|	[0.025	0.975]
const	2.0554	0.019	108.830	0.000	2.018	2.092
x1	0.2970	0.006	48.237	0.000	0.285	0.309

Omnibus:	1.085	Durbin-Watson:	2.113
Prob(Omnibus):	0.581	Jarque-Bera (JB):	1.106
Skew:	0.002	Prob(JB):	0.575
Kurtosis:	2.828	Cond. No.	3.69

https://github.com/foxtrotmike/CS909/blob/master/regression_residuals.ipynb

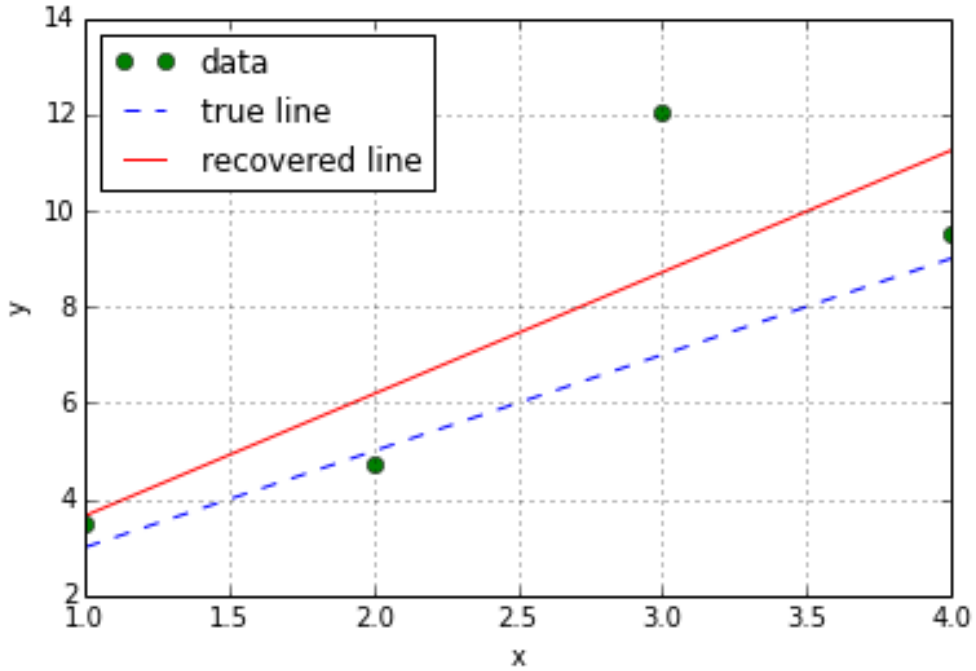
Questions

- If two variables are unrelated
  - What would be their regression coefficients?
- If two variables are positively related
  - What would be their regression coefficient?
- Would the regression coefficient be the same (in general) if I go in the reverse direction?
  - $y = w_{xy}x + b_{xy} + \epsilon_{xy}$
  - $x = w_{yx}y + b_{yx} + \epsilon_{yx}$
- Would the error be correlated with the independent variable?



Problems with least squares solution

- No regularization
  - We want the parameters to change slightly with change in the input data

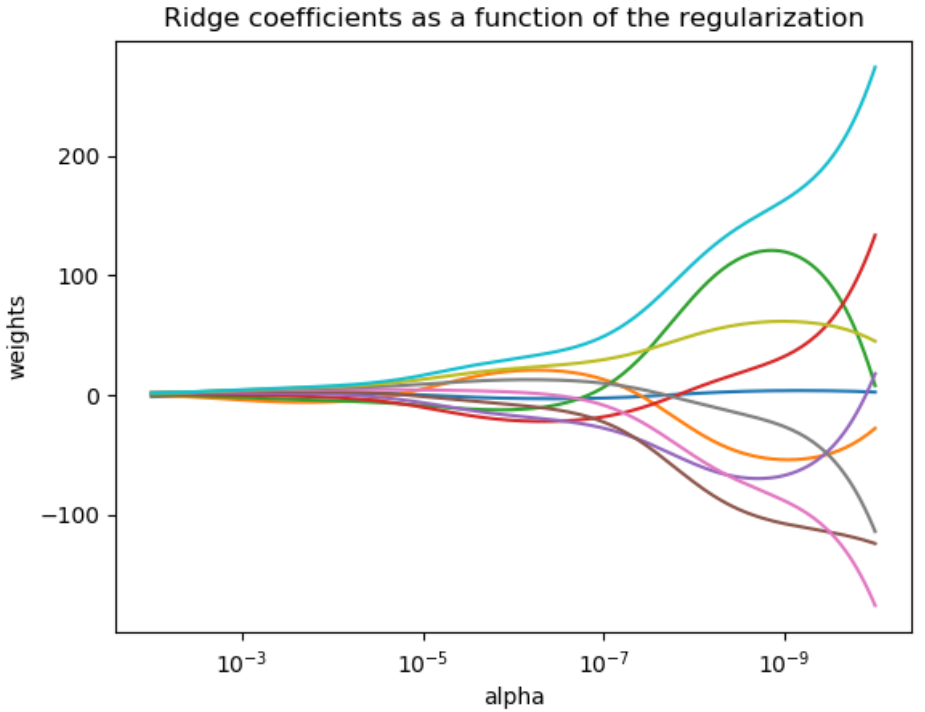


Regularized Least Squares Regression

- Ridge Regression
  - Ridge regression addresses some of the problems of Ordinary Least Squares by imposing a penalty on the size of the coefficients. The ridge coefficients minimize a penalized residual sum of squares:

$$\min_w ||Xw - y||_2^2 + \alpha||w||_2^2$$

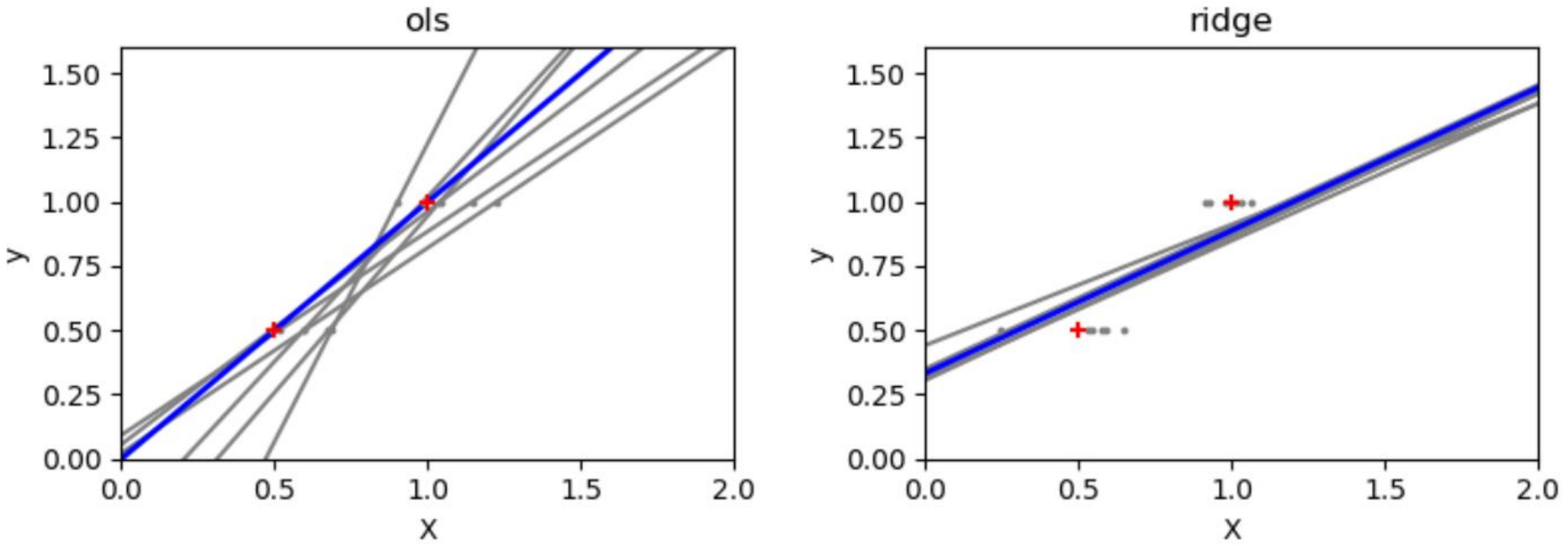
    - What is the role of the hyperparameter alpha?



Coding

```
from sklearn import linear_model
ridge = linear_model.Ridge(alpha=a)
```

OLS vs. Ridge Regression



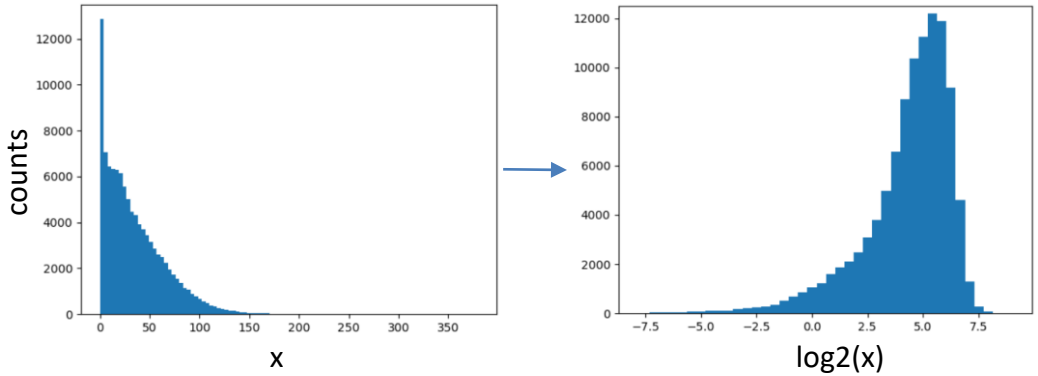
- Ridge regression is basically minimizing a penalised version of the least-squared function. The penalising shrinks the value of the regression coefficients. Despite the few data points in each dimension, the slope of the prediction is much more stable and the variance in the line itself is greatly reduced, in comparison to that of the standard linear regression

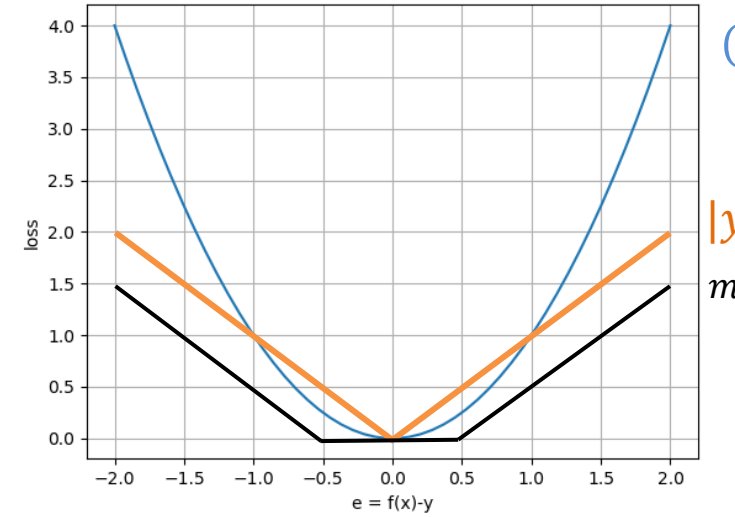
https://github.com/foxtrotmike/CS909/blob/master/regression.ipynb

https://scikit-learn.org/stable/auto_examples/linear_model/plot_ols_ridge_variance.html#sphx-glr-auto-examples-linear-model-plot-ols-ridge-variance-py

Issues

- Both OLS and Ridge Regression are linear models
  - Only able to predict “linear” relationships between input and output variables
- Linear regression assumes that the relationship between independent and dependent variables is linear and the error residuals should be normally distributed (link to code)
  - So before applying it, you should check this by plotting the histogram of the residuals
  - Otherwise, you can transform the variable (e.g., through box-cox or other transforms) or use generalized linear models or other strategies
- Due to squaring of the error of each data point in the loss function, both OLS and ridge regression can be very sensitive to outliers
- We can reduce the impact of outliers by using softer penalties
  - Linear penalty
  - Epsilon insensitive loss





$(y' - y)^2$

$|y' - y|$

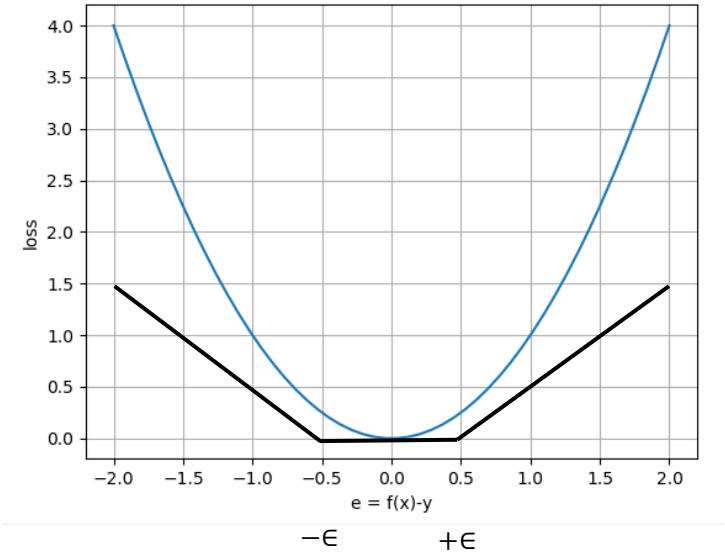
$max(0, |f(\boldsymbol{x}) - y| - \epsilon)$

From ∈-insensitive loss to SVR

- ∈-insensitive loss

$$l(f(x), y) = max(0, |f(\boldsymbol{x}) - y| - \in)$$

- Support Vector Regression
  - Representation
    - $f(\boldsymbol{x}) = \boldsymbol{w^T x} + b$
  - Evaluation
    - Regularization: $\boldsymbol{w^T w}$
    - Error: Epsilon insensitive loss
  - Optimization Problem



$$\min_{\boldsymbol{w},\boldsymbol{b}} \frac{1}{2}\boldsymbol{w^T w} + \frac{C}{\boldsymbol{N}} \sum_{\boldsymbol{i=1}}^{\boldsymbol{N}} max(0, |f(\boldsymbol{x_i}) - y_i| - \in)$$

Non-linear Regression with SVR

- Using Representer Theorem & Kernel Trick

$$\min_{\boldsymbol{w},b} \frac{1}{2}\boldsymbol{w^T w} + \frac{C}{\boldsymbol{N}}\sum_{\boldsymbol{i=1}}^{\boldsymbol{N}} \boldsymbol{max}(0, |f(\boldsymbol{x_i}) - y_i| - \in)$$

Representer Theorem: $\boldsymbol{w} = \sum_{i=1}^{N} \alpha_i \boldsymbol{x}_i$

$$f(\boldsymbol{x}) = \boldsymbol{w}^T x + b = \sum_{i=1}^{N} \alpha_i \boldsymbol{x}_i^T \boldsymbol{x} + b$$

$$\min_{\boldsymbol{\alpha},b} \frac{1}{2}\sum_{i,j=1}^{N} \alpha_i \alpha_j \, \boldsymbol{x}_i{}^{\boldsymbol{T}} \boldsymbol{x}_j + \frac{C}{\boldsymbol{N}}\sum_{\boldsymbol{i=1}}^{\boldsymbol{N}} \boldsymbol{max}\left(0, \left|\sum_{j=1}^{N} \alpha_i \boldsymbol{x}_i^T \boldsymbol{x}_j + b - y_i\right| - \in\right)$$

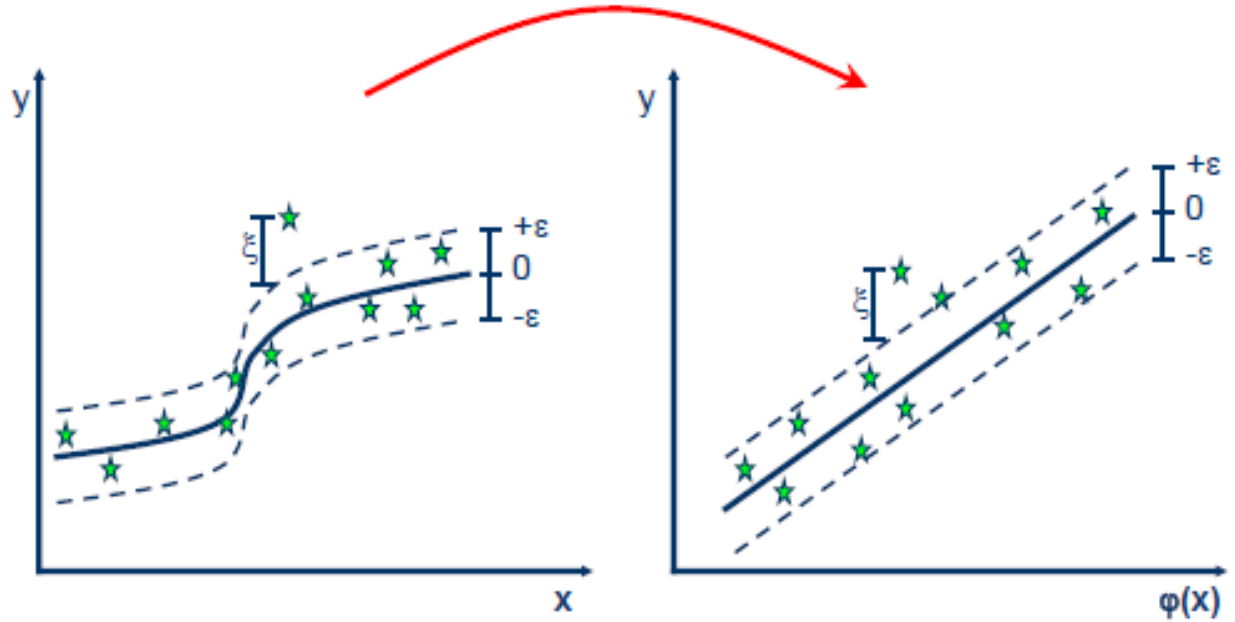
Kernel Trick: Replacing $\boldsymbol{x}_i{}^{\boldsymbol{T}} \boldsymbol{x}_j$ with $k(x_i, x_j)$

$$f(\boldsymbol{x}) = \boldsymbol{w}^T x + b = \sum_{i=1}^{N} \alpha_i \boldsymbol{x}_i^T \boldsymbol{x} + b = \sum_{i=1}^{N} k(x_i, x) + b$$

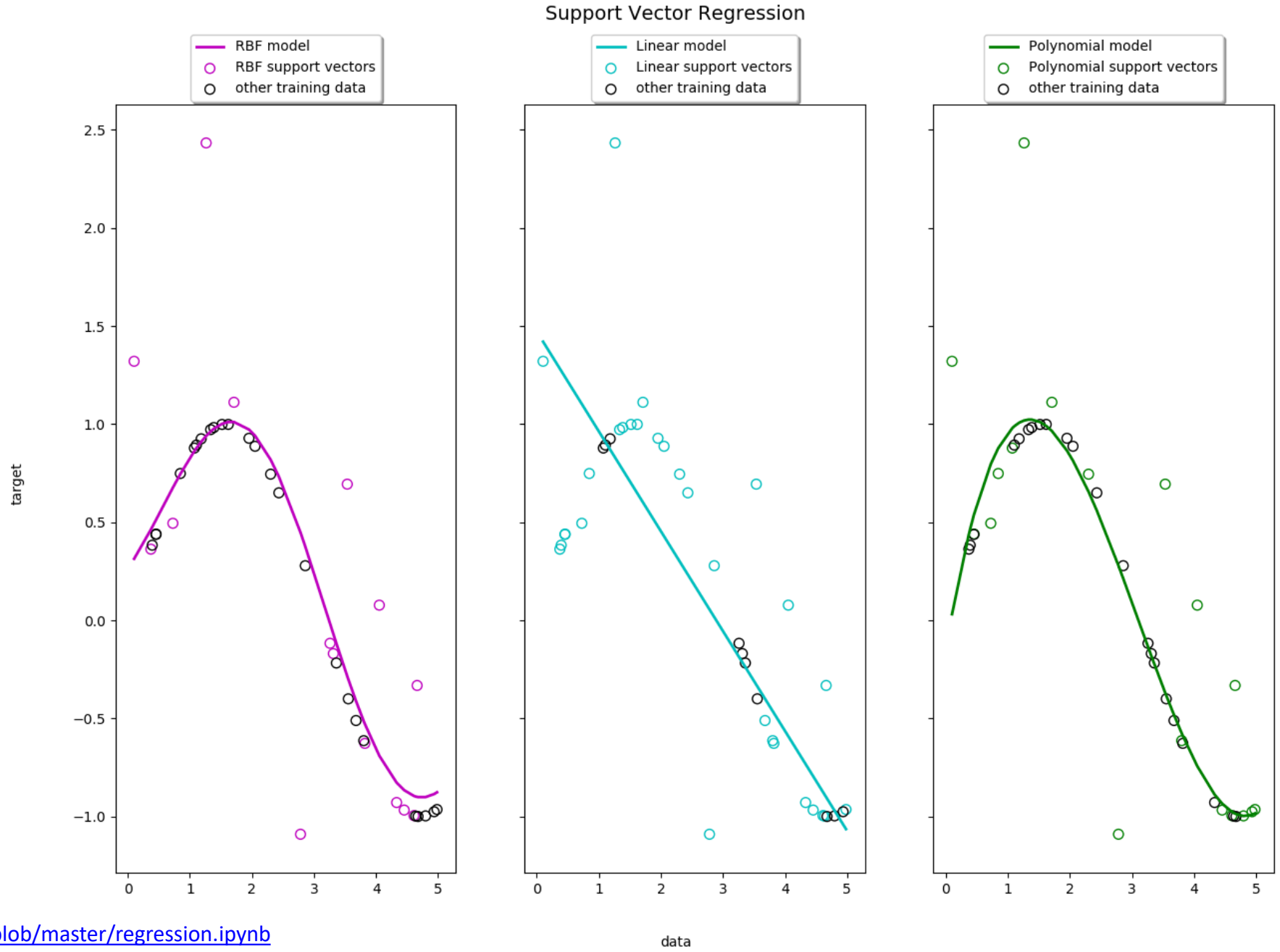
$$\min_{\boldsymbol{\alpha},b} \frac{1}{2}\sum_{i,j=1}^{N} \alpha_i \alpha_j \, k(\boldsymbol{x_i}, \boldsymbol{x_j}) + \frac{C}{\boldsymbol{N}}\sum_{\boldsymbol{i=1}}^{\boldsymbol{N}} \boldsymbol{max}\left(0, \left|\sum_{j=1}^{N} k(\boldsymbol{x_i}, \boldsymbol{x_j}) + b - y_i\right| - \in\right)$$

Kernels in SVR

- In the kernel space the SVR is fitting a line which corresponds to an arbitrary curve in the original feature space



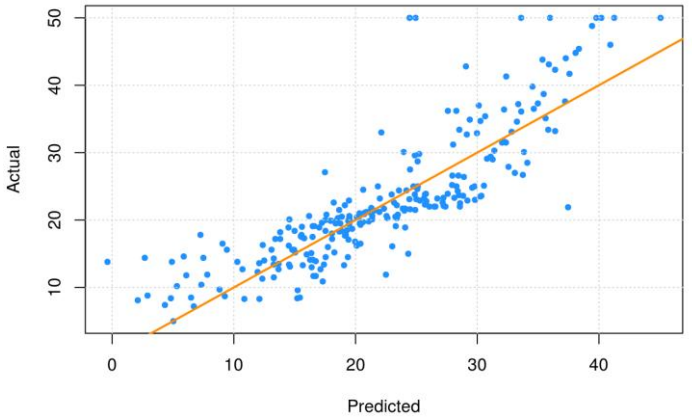
Kernels	Functions	Parameters
Linear	$< x, x_i >$	
Polynomial	$(< x, x_i > +1)^d$	d
Radial basis function	$\exp\left(-\|x-x_i\|^2/2\sigma^2\right)$	σ
Sigmoid	$\tanh(b<x, x_i>+c)$	b, c



See: https://github.com/foxtrotmike/CS909/blob/master/regression.ipynb
https://scikit-learn.org/stable/auto_examples/svm/plot_svm_regression.html

Cross-validation and performance metrics

- Evaluation protocols for hyperparameter selection remain the same as in classification
- How to calculate how good we are doing?
  - Measure error or correlation between true and predicted outputs for different examples in a test set
- Error Measures
  - Sum of squared errors
  - Mean squared error
  - Mean Absolute error
  - Root mean square error
  - R^2 score
- Pearson and Spearman Correlation Coefficients



$$\mathrm{MSE}(y, \hat{y}) = \frac{1}{n_{\text{samples}}} \sum_{i=0}^{n_{\text{samples}}-1} (y_i - \hat{y}_i)^2$$

$$\mathrm{MAE}(y, \hat{y}) = \frac{1}{n_{\text{samples}}} \sum_{i=0}^{n_{\text{samples}}-1} |y_i - \hat{y}_i|$$

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}}$$

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2}$$

$$r = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n} (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^{n} (y_i - \bar{y})^2}}$$

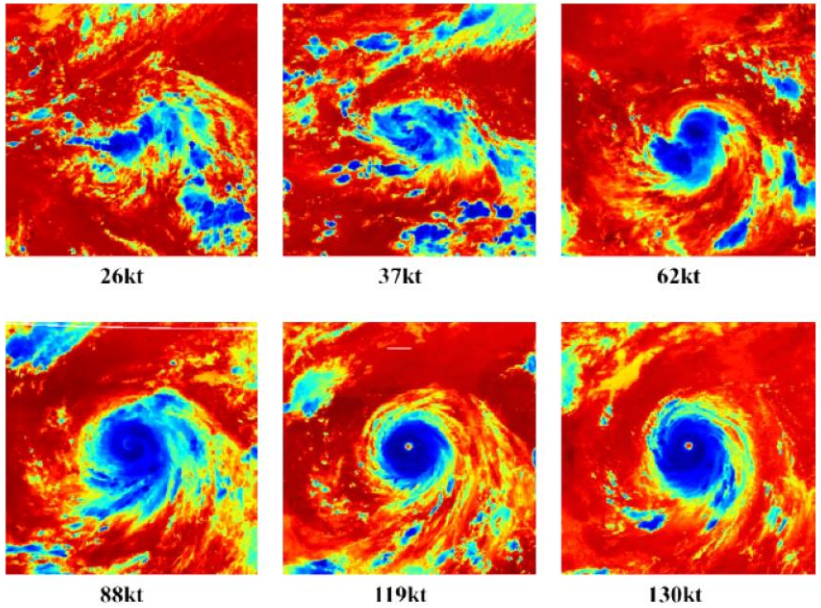
https://scikit-learn.org/stable/modules/model_evaluation.html

Practical Application

Hurricane Intensity Estimation

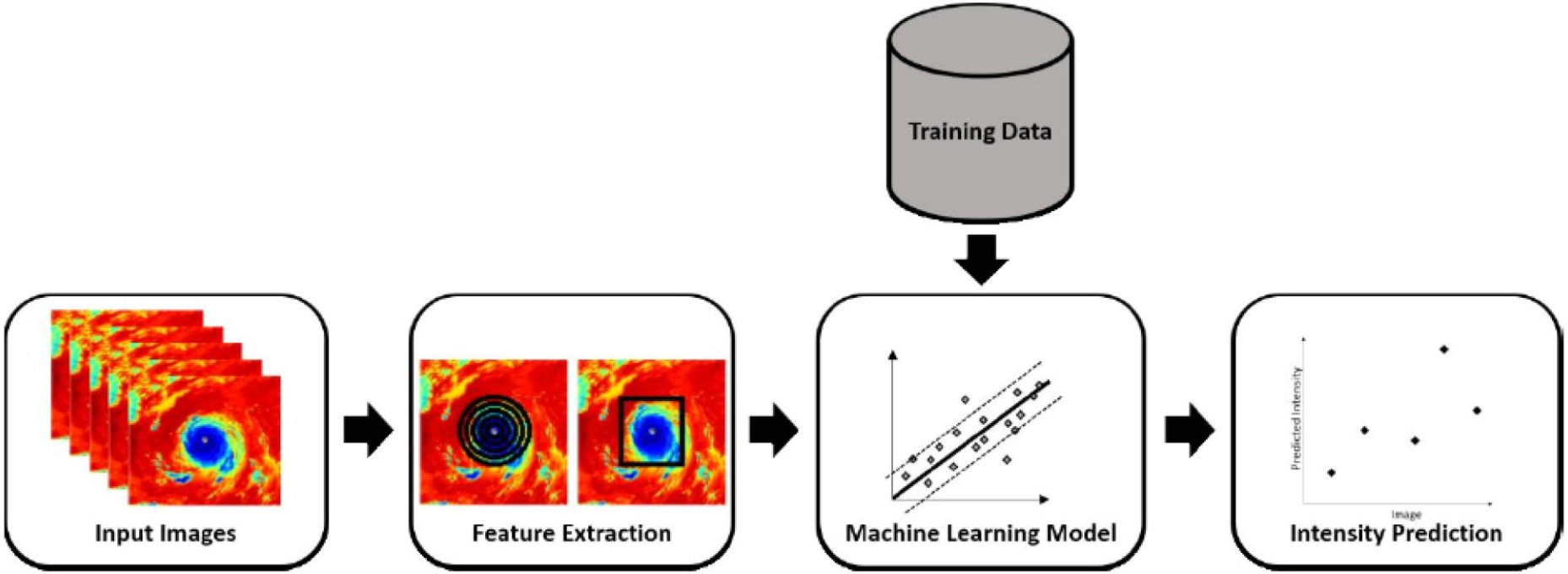
Input: Infrared Satellite Images of Hurricanes

Output: Maximum Sustained Windspeed in knots



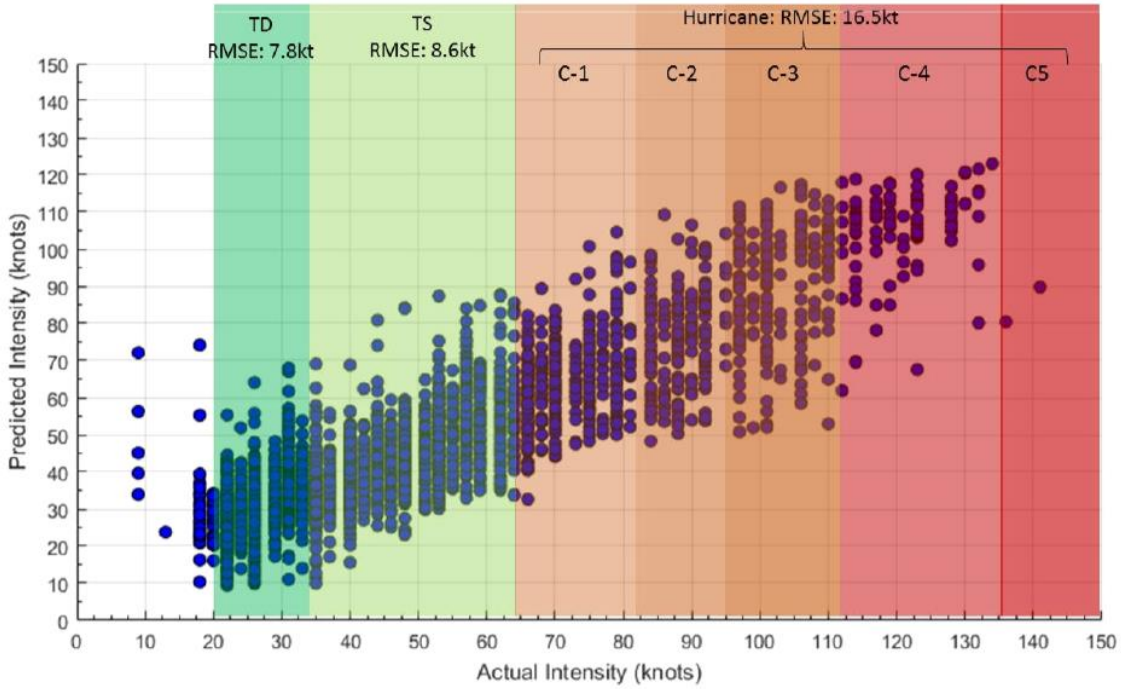
https://en.wikipedia.org/wiki/Hurricane_hunters

PHURIE ML Pipeline

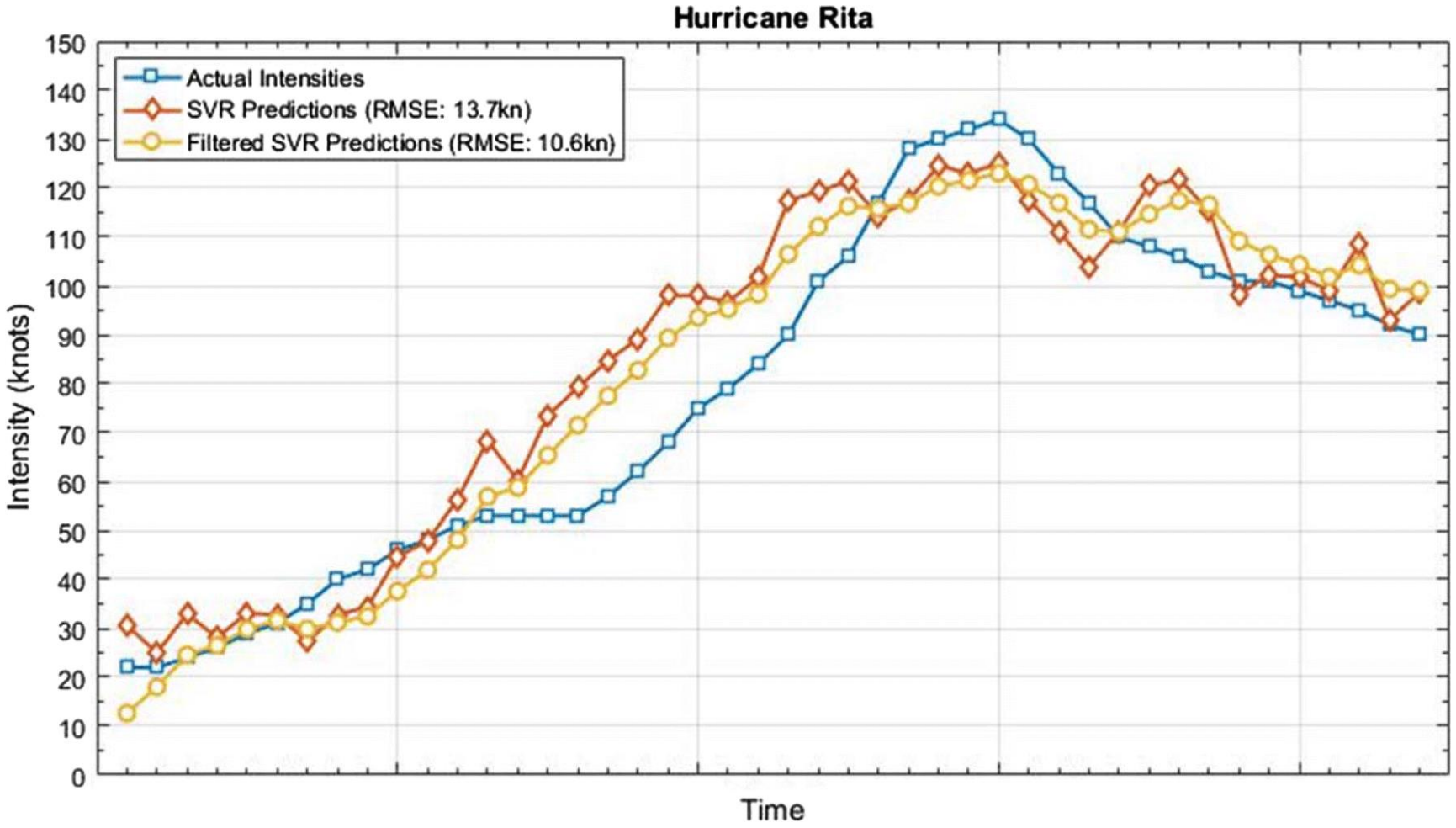


PHURIE: Hurricane Intensity Estimation from Infrared Satellite Imagery using Machine Learning, Amina Asif, Muhammad Dawood, Bismillah Jan, Javaid Khurshid, Mark DeMaria, and Fayyaz ul Amir Afsar Minhas, in Neural Computing and Applications, DOI: http://dx.doi.org/10.1007/s00521-018-3874-6, 2018. (Paper)

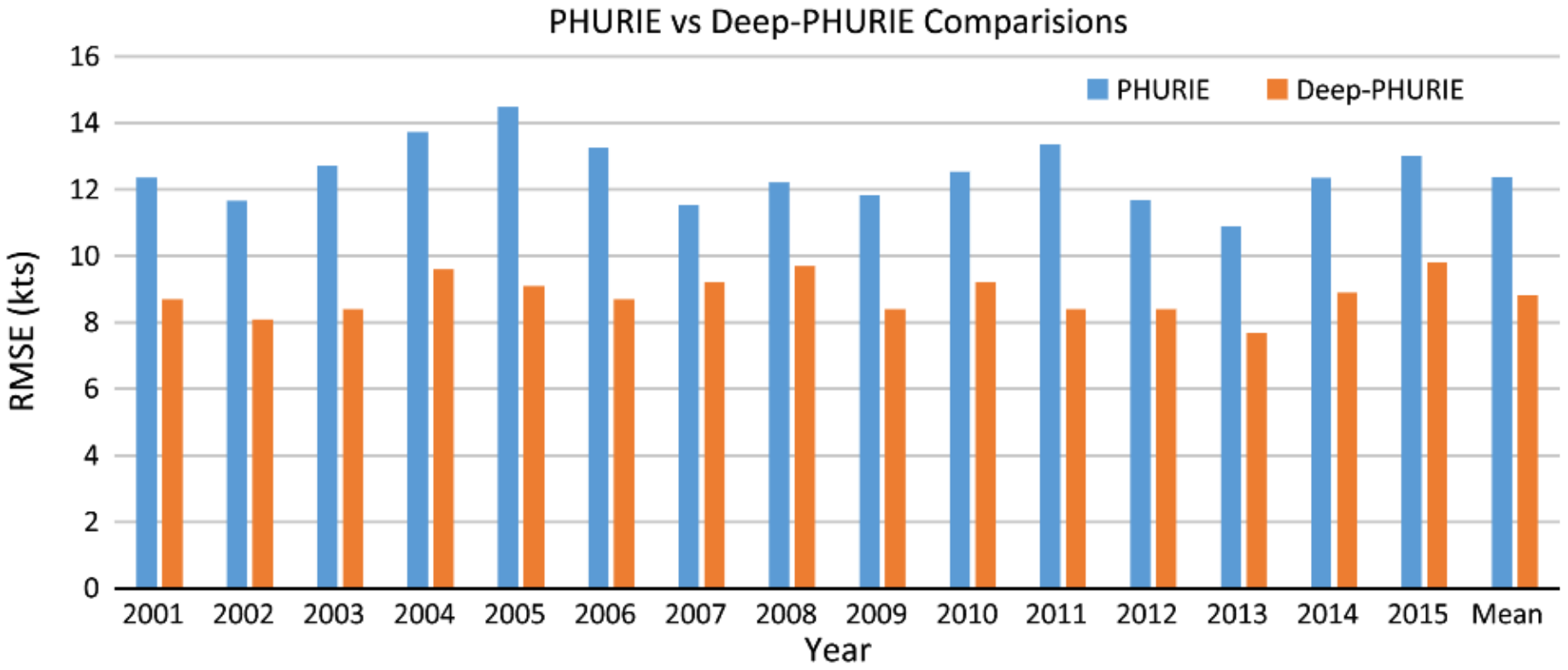
Practical Application



Method	Mean RMSE (kt)	Mean RMSE after smoothing
PHURIE: SVR	**11.2**	**9.5**
PHURIE: OLS	12.8	10.5
PHURIE: BPNN	12.0	10.1
PHURIE: XGBoost	11.3	9.8
Baseline predictor (mean)	24.3	–


Hurricane Rita
Actual Intensities
SVR Predictions (RMSE: 13.7kn)
Filtered SVR Predictions (RMSE: 10.6kn)
Intensity (knots)
Time
0
10
20
30
40
50
60
70
80
90
100
110
120
130
140
150

Extension: Deep PHURIE



Deep-PHURIE: Deep Learning based Hurricane Intensity Estimation from Infrared Satellite Imagery, M. Dawood, A. Asif and Fayyaz Minhas, in Neural Computing and Applications. pp. DOI: 10.1007/s00521-019-04410-7, July 2019.

Required

- Reading:
  - Section 13.10 in Alpaydin 2010
  - http://ciml.info/ Linear Models
  - Very useful: Regression with linear models
    - https://scikit-learn.org/stable/modules/linear_model.html
  - Code: **Scikit-learn**, https://github.com/NICTA/revrand
  - StatsModel Regression Exercises: https://www.statsmodels.org/dev/examples/index.html