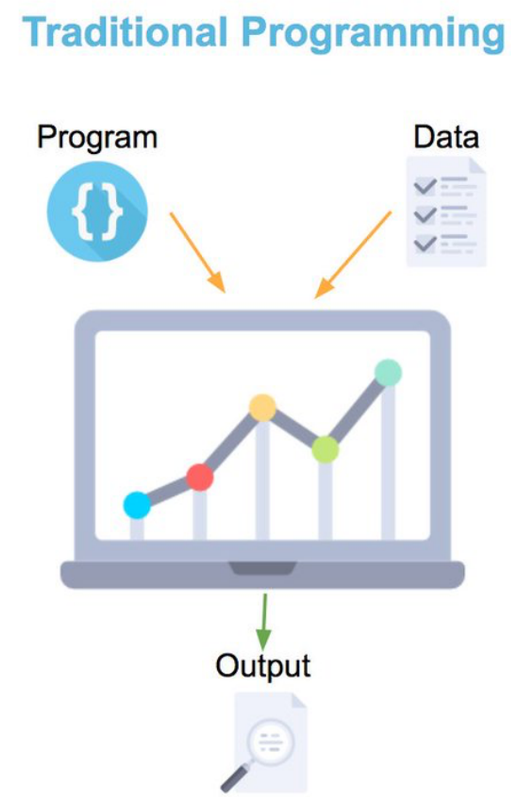
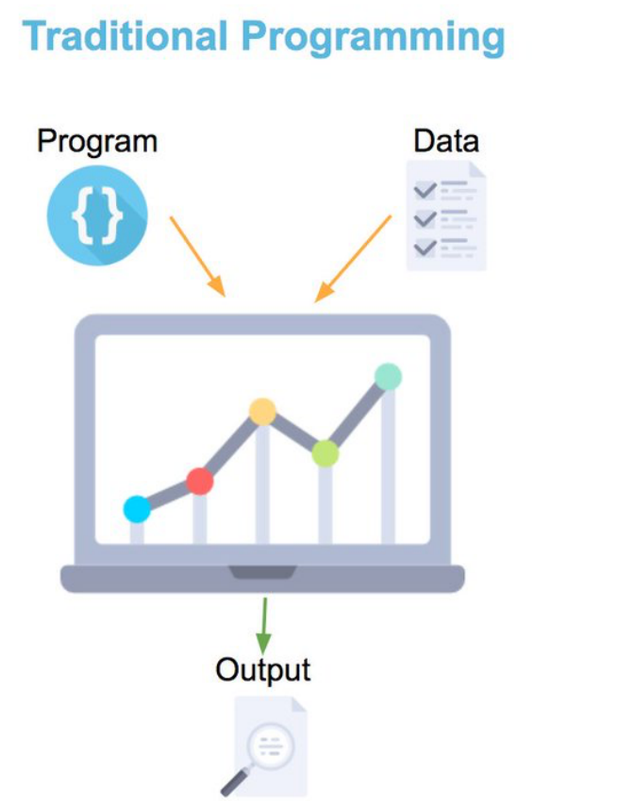
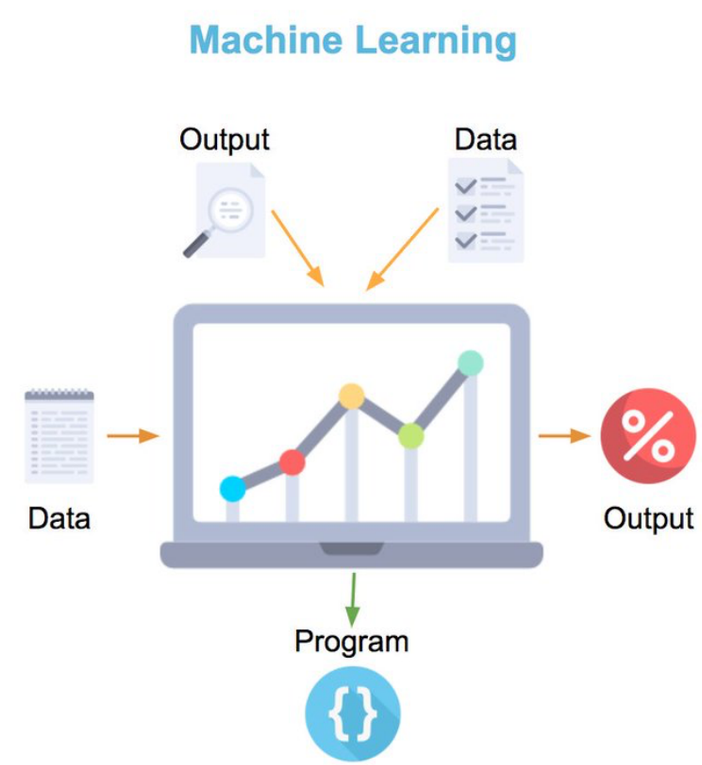
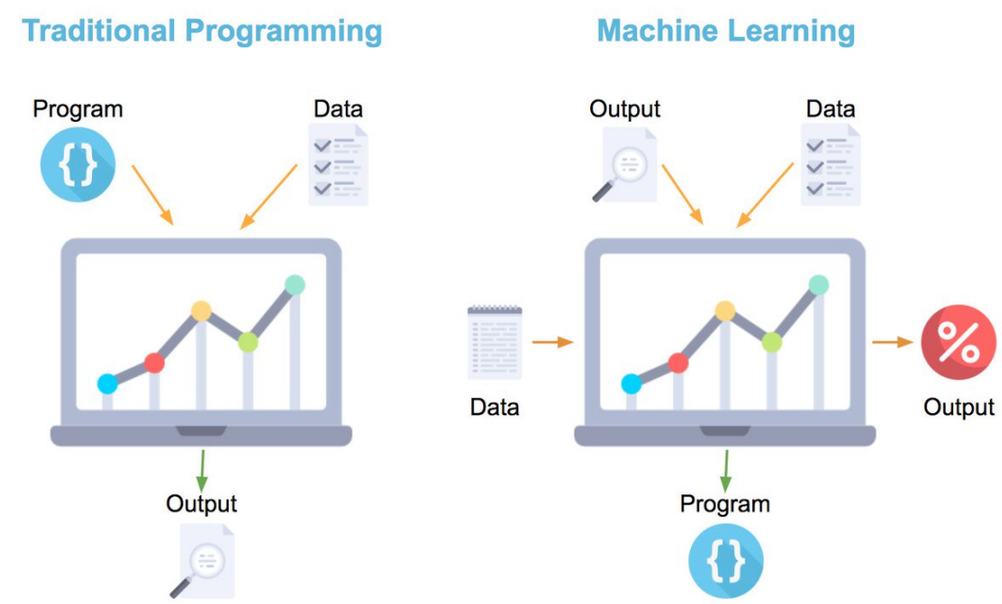
INTRODUCTION TO MACHINE LEARNING

MACHINE LEARNING FOR PROGRAMMERS

- (Machine) Programming: define each step
  - ```
    factorial = 1
    for i in range(2, n+1):
        factorial = factorial * i
    ```

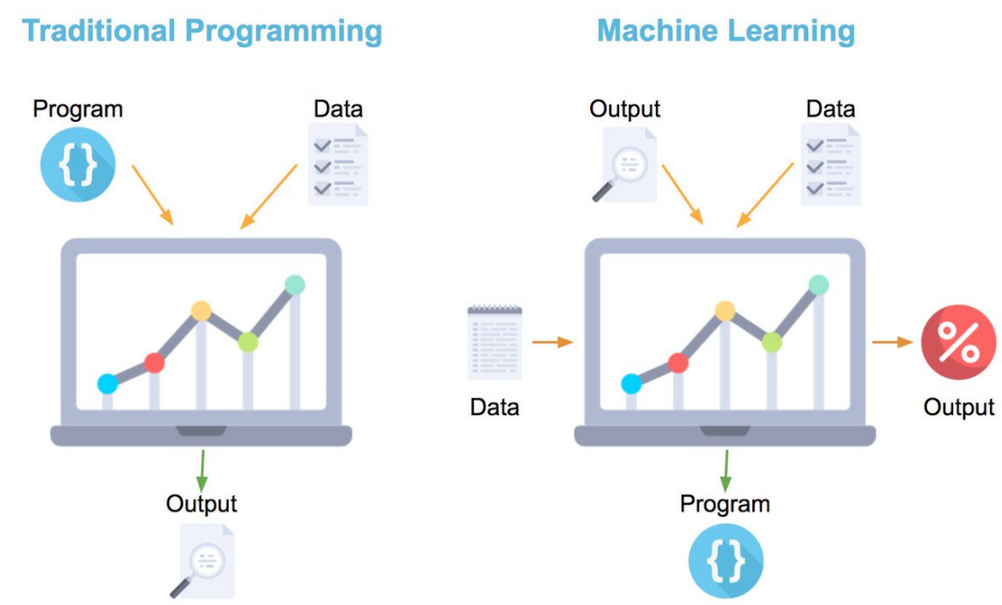

    ```

MACHINE LEARNING FOR PROGRAMMERS

- (Machine) Programming: define each step
  - ```
    factorial = 1
    for i in range(2, n+1):
        factorial = factorial * i
    ```
- Machine learning: we do not define the steps (because we cant)
  - We provide examples
  - We create a framework, and the optimising algorithm
  - Then the machine 'learns', alas we optimize
- Learning: the program is created using good examples




    ```

MACHINE LEARNING FOR PHYSICISTS

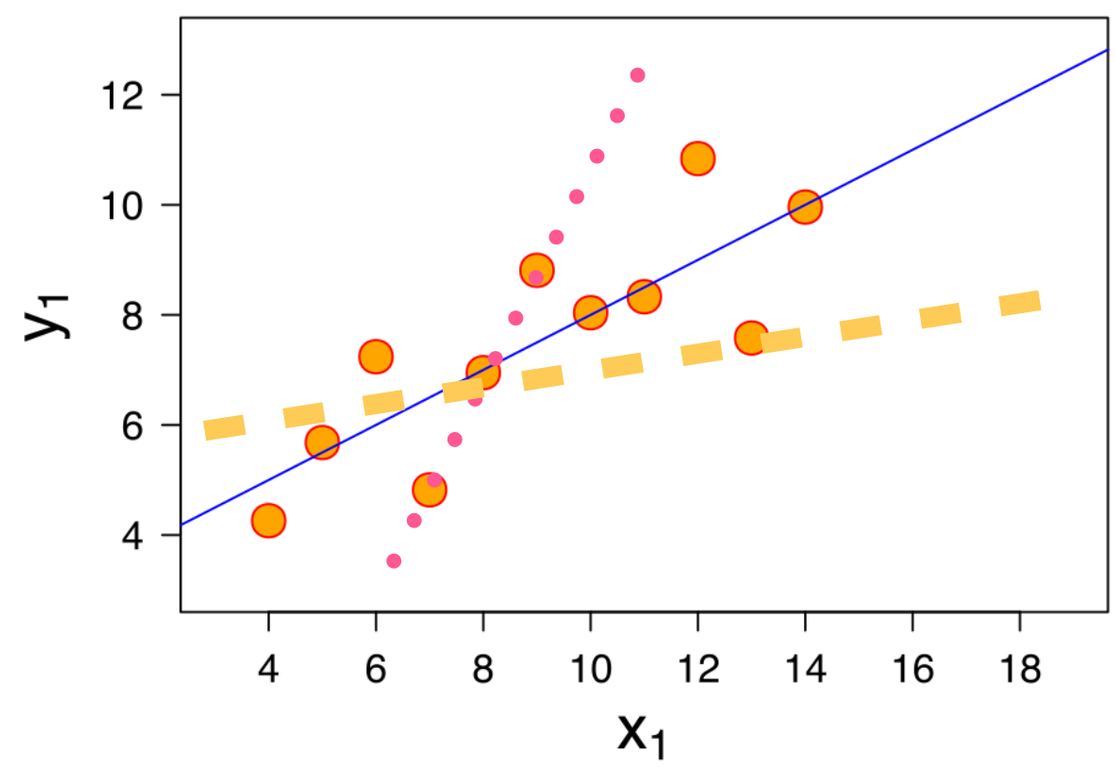


- Programs are functions. They have inputs, and outputs:
  - y = F(x)
- Programming: executes a mapping (function) we describe, step by step

MACHINE LEARNING FOR PHYSICISTS

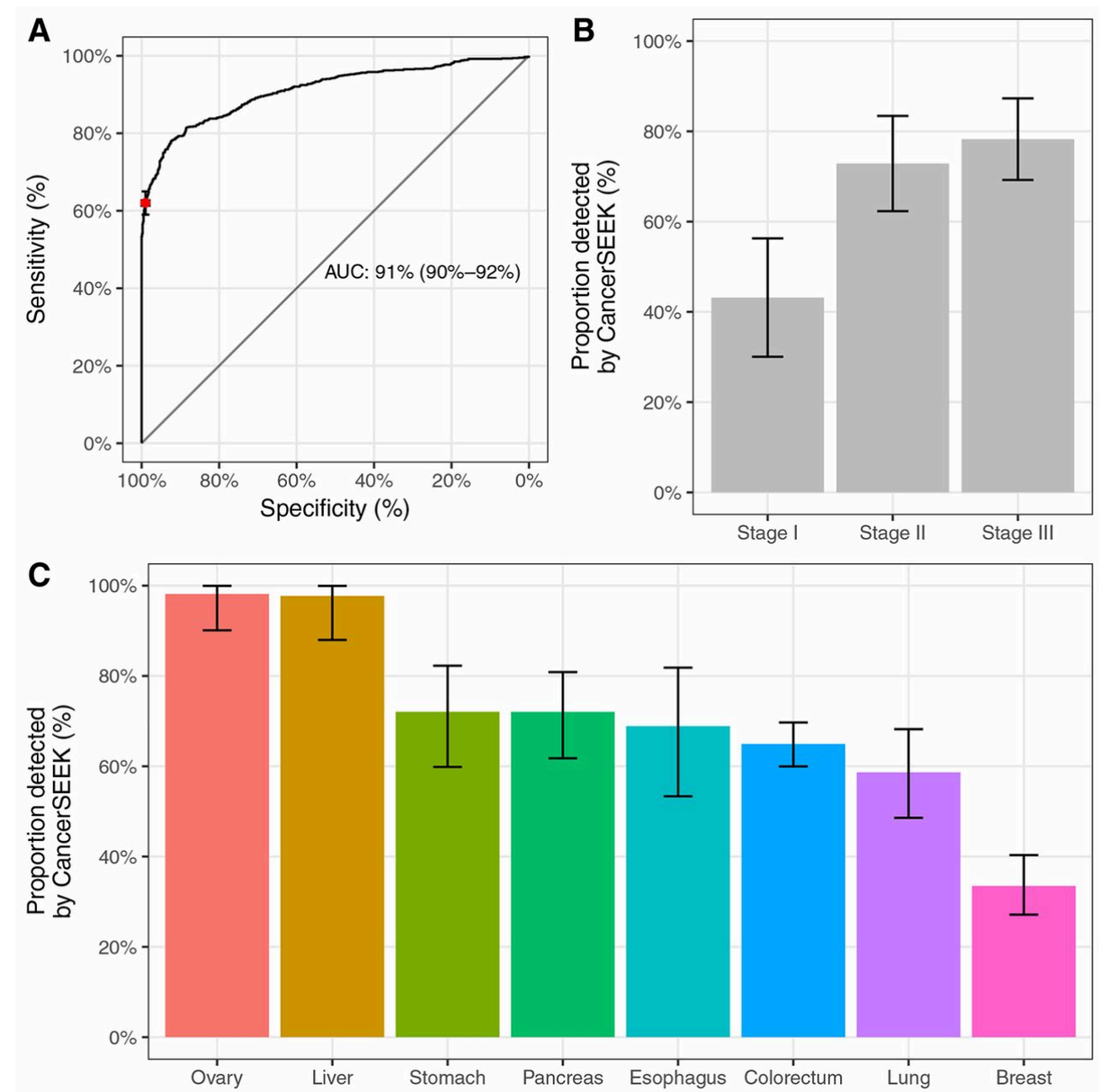


- Programs are functions. They have inputs, and outputs:
  - y = F(x)
- Programming: executes a mapping (function) we describe, step by step
- Machine learning is functions approximation, fitting!
  - We prove the (parametric) function:
    - y = ax + b
  - "Learning:" We fit the parameters (a,b) of the function to measured data points (x1,y1) points



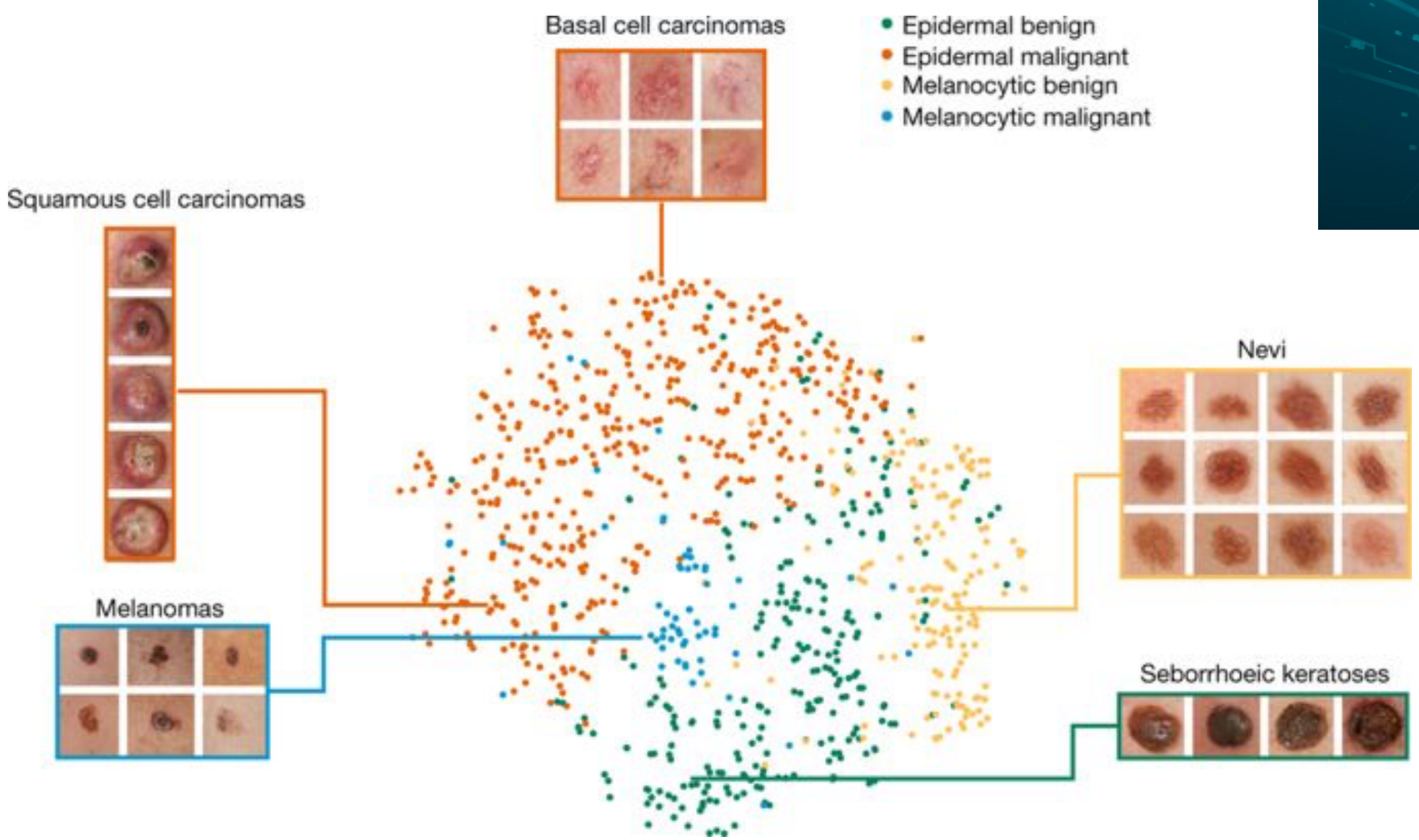
WHAT IS MACHINE LEARNING?

- Examples:
  - Estimate whether a patient has cancer
  - based on the protein levels and mutations of genes in their blood
  - Cohen, Joshua D., et al. "Detection and localization of surgically resectable cancers with a multi-analyte blood test." Science (2018).



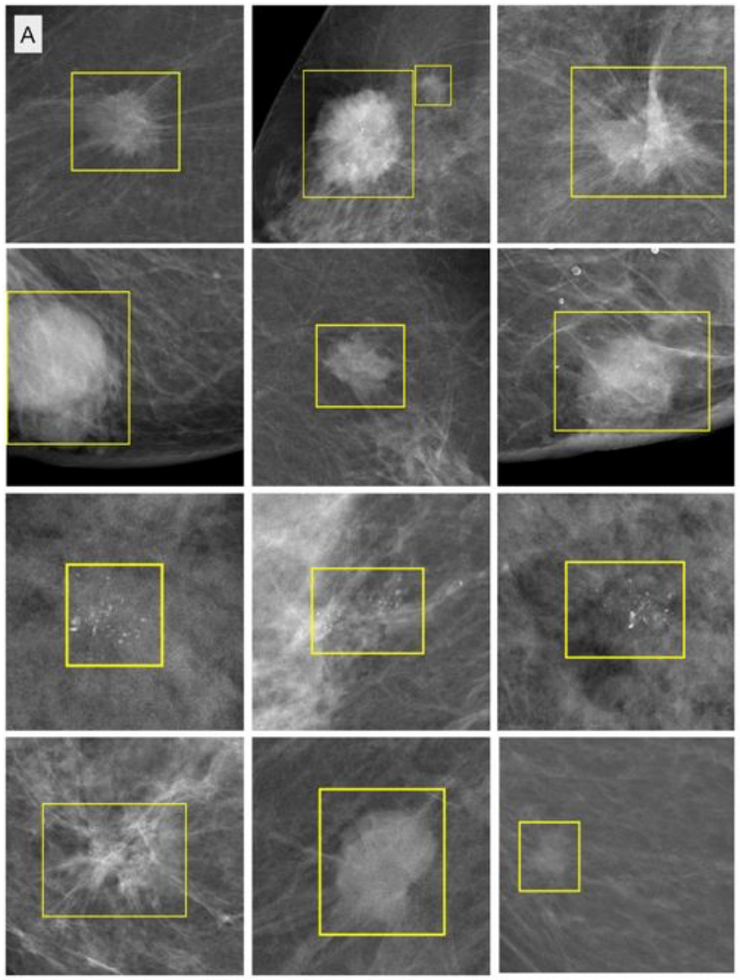
WHAT IS MACHINE LEARNING?

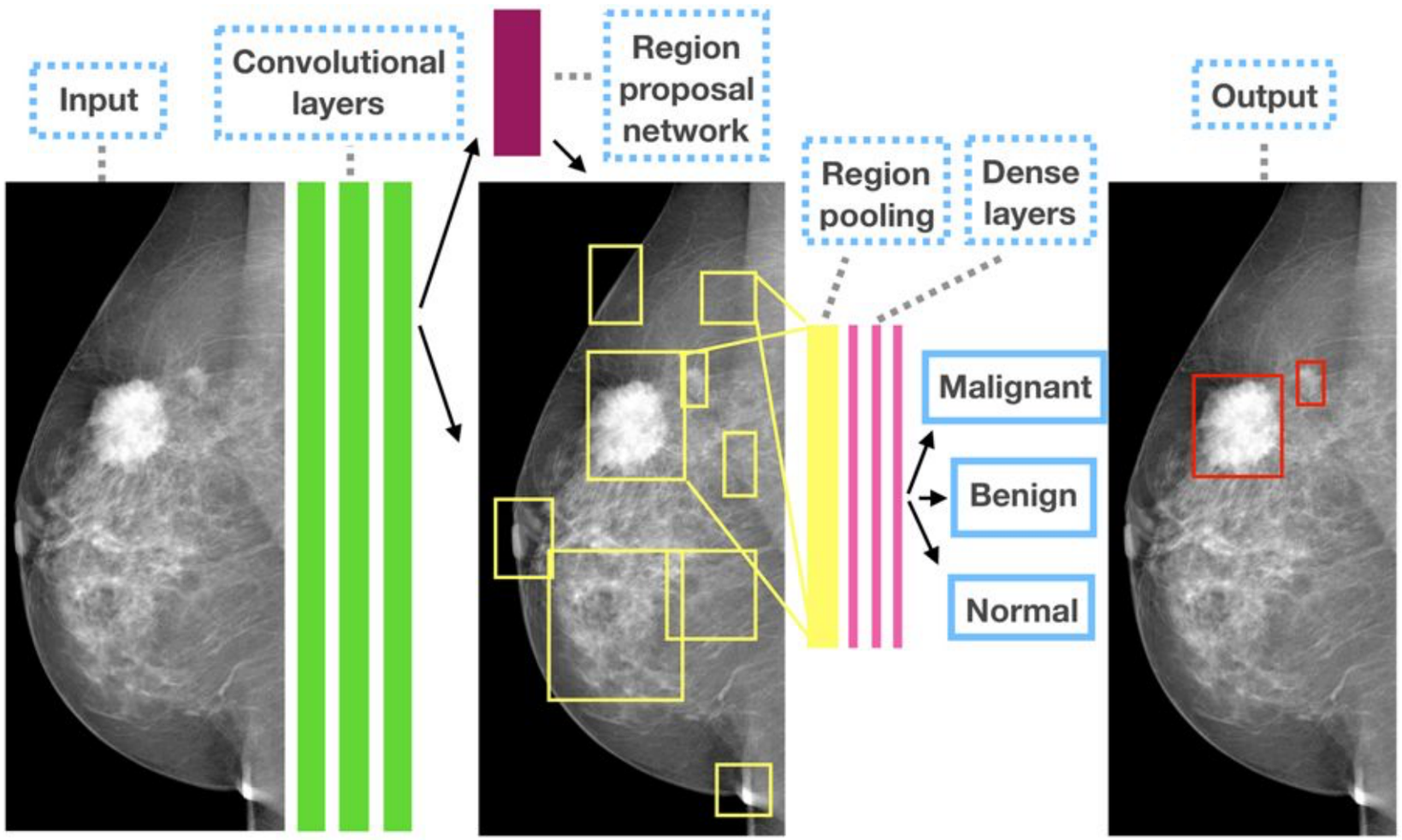
- Recognise if a patient has cancer, based on medical imaging data (x-ray, CT, MRI, ultrasound, cytology, etc)
- Esteva, et al. "Dermatologist-level classification of skin cancer with deep neural networks." Nature



WHAT IS MACHINE LEARNING?

- Recognise if a patient has cancer, based on medical imaging data (x-ray, CT, MRI, ultrasound, cytology, etc)
- Ribli et al. "Detecting and classifying lesions in mammograms with Deep Learning." Scientific reports





MACHINE LEARNING IS PREDICTIVE MODELLING

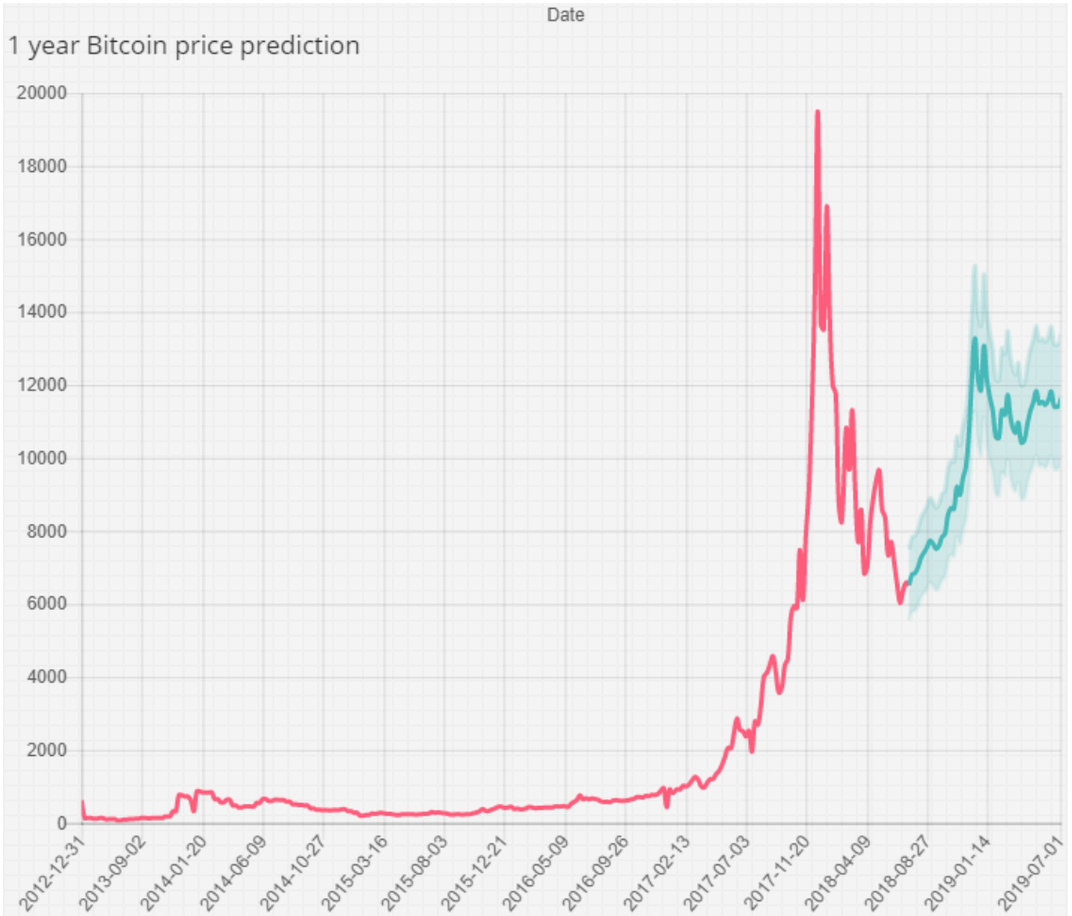
- Scientific, physics models: “understand”, describe the laws of nature, “simple” relationship between observable quantities (equations)

$$E = mc^2$$

MACHINE LEARNING IS PREDICTIVE MODELLING

- Scientific, physics models: “understand”, describe the laws of nature, “simple” relationship between observable quantities (equations)
- Predictive modelling
  - y = F(x) unknown function approximation, x: easy to measure, y: would be useful to estimate
  - The accuracy is essential, understanding is secondary
  - Examples: targetet advertisement, stock-predictions, Go-chess next best step “prediction”, sound language processing, image recognition
  - Useful in scientific research too!

$$E = mc^2$$



MACHINE LEARNING IS PREDICTIVE MODELLING

- Scientific, physics models: “understand”, describe the laws of nature, “simple” relationship between observable quantities (equations)
- Predictive modelling
  - y = F(x) unknown function approximation, x: easy to measure, y: would be useful to estimate
  - The accuracy is essential, understanding is secondary
  - Examples: targetet advertisement, stock-predictions, Go-chess next best step “prediction”, sound language processing, image recognition
  - Useful in scientific research too!

$$E = mc^2$$

2			4	5	
5		4			1
		5		2	
	1		5		4
		4			2
4	5		1		

MACHINE LEARNING IS PREDICTIVE MODELLING

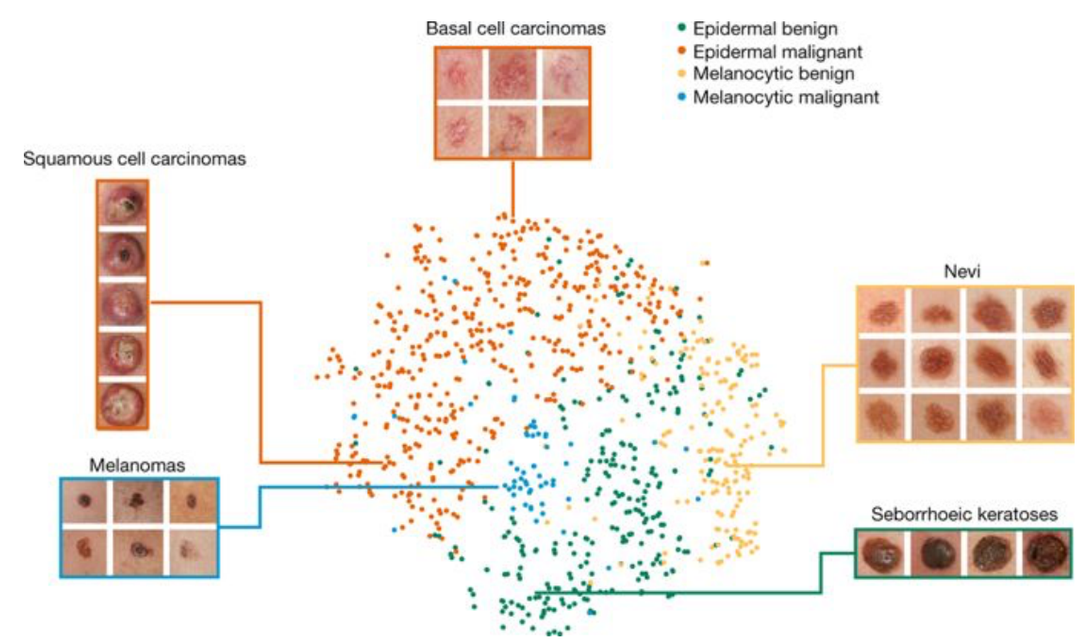
- Scientific, physics models: “understand”, describe the laws of nature, “simple” relationship between observable quantities (equations)
- Predictive modelling
  - y = F(x) unknown function approximation, x: easy to measure, y: would be useful to estimate
  - The accuracy is essential, understanding is secondary
  - Examples: targetet advertisement, stock-predictions, Go-chess next best step “prediction”, sound language processing, image recognition
  - Useful in scientific research too!

$$E = mc^2$$

MACHINE LEARNING IS PREDICTIVE MODELLING

- Scientific, physics models: “understand”, describe the laws of nature, “simple” relationship between observable quantities (equations)
- Predictive modelling
- y = F(x) unknown function approximation, x: easy to measure, y: would be useful to estimate
- The accuracy is essential, understanding is secondary
- Examples: targetet advertisement, stock-predictions, Go-chess next best step “prediction”, sound language processing, image recognition
- Useful in scientific research too!

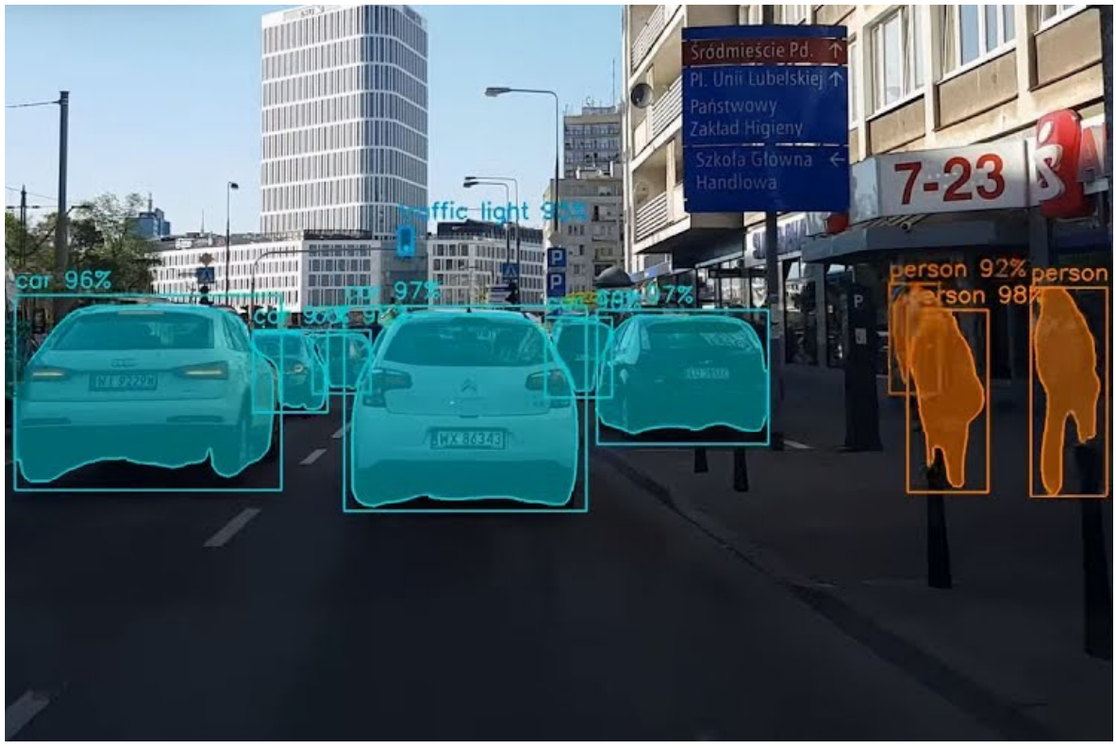
$$E = mc^2$$



MACHINE LEARNING IS PREDICTIVE MODELLING

- Scientific, physics models: “understand”, describe the laws of nature, “simple” relationship between observable quantities (equations)
- Predictive modelling
- y = F(x) unknown function approximation, x: easy to measure, y: would be useful to estimate
- The accuracy is essential, understanding is secondary
- Examples: targetet advertisement, stock-predictions, Go-chess next best step “prediction”, sound language processing, image recognition
- Useful in scientific research too!

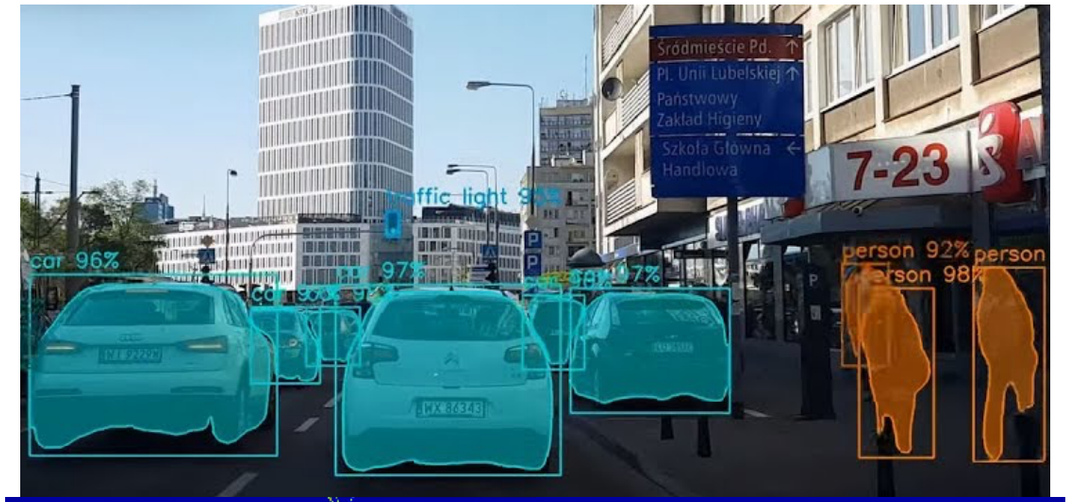
$$E = mc^2$$

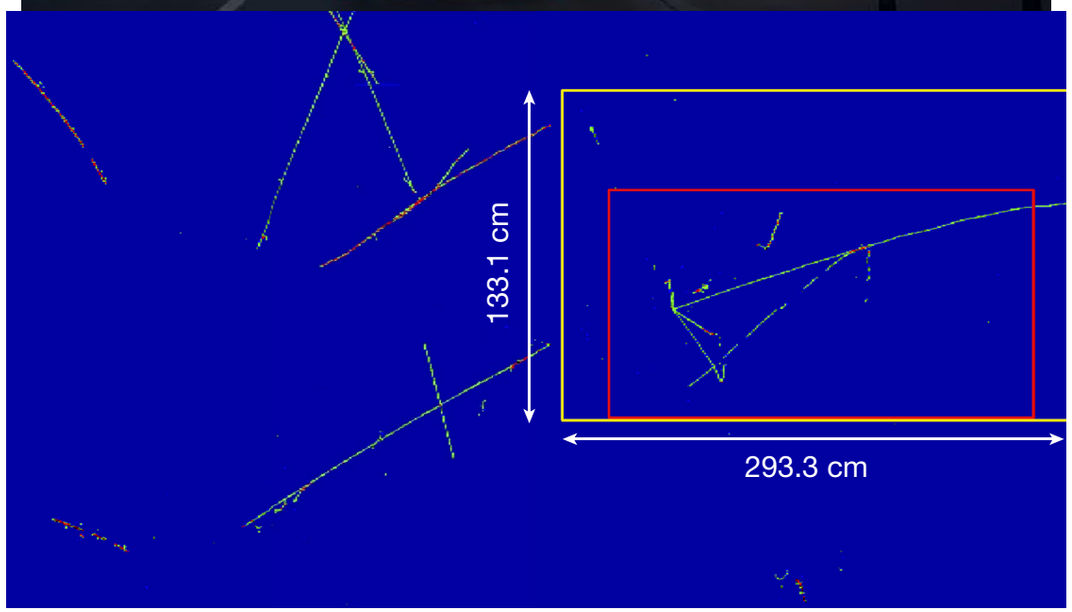


MACHINE LEARNING IS PREDICTIVE MODELLING

- Scientific, physics models: “understand”, describe the laws of nature, “simple” relationship between observable quantities (equations)
- Predictive modelling
  - y = F(x) unknown function approximation, x: easy to measure, y: would be useful to estimate
  - The accuracy is essential, understanding is secondary
  - Examples: targetet advertisement, stock-predictions, Go-chess next best step “prediction”, sound language processing, image recognition
  - Useful in scientific research too!

$$E = mc^2$$

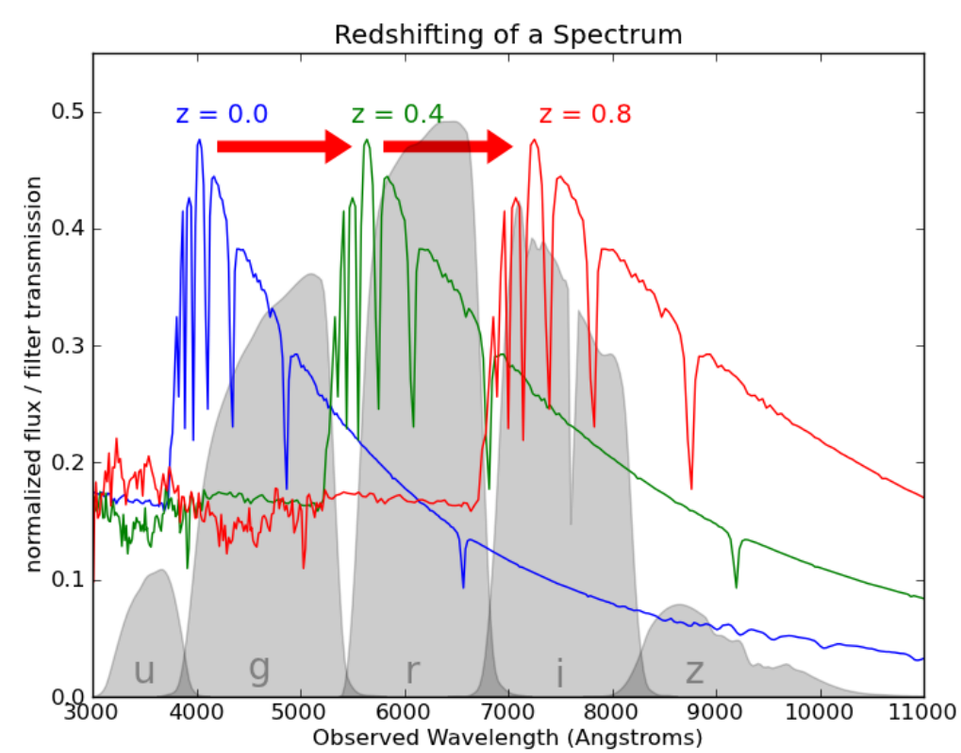




MACHINE LEARNING IS PREDICTIVE MODELLING

- Scientific, physics models: “understand”, describe the laws of nature, “simple” relationship between observable quantities (equations)
- Predictive modelling
- y = F(x) unknown function approximation, x: easy to measure, y: would be useful to estimate
- The accuracy is essential, understanding is secondary
- Examples: targetet advertisement, stock-predictions, Go-chess next best step “prediction”, sound language processing, image recognition
- Useful in scientific research too!

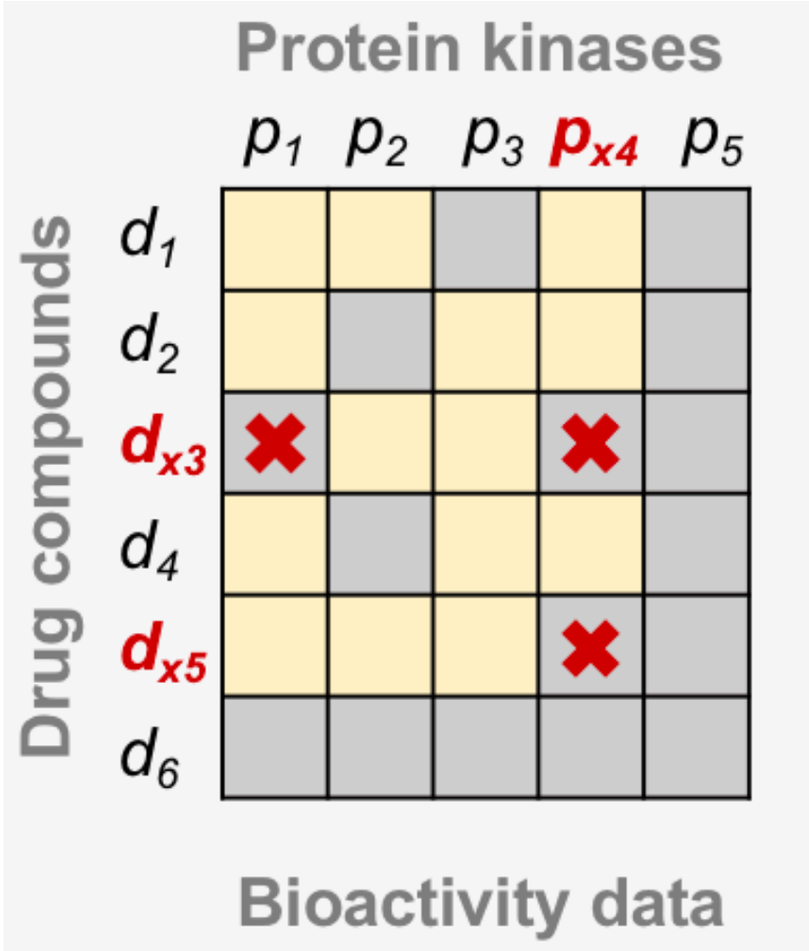
$$E = mc^2$$



MACHINE LEARNING IS PREDICTIVE MODELLING

- Scientific, physics models: “understand”, describe the laws of nature, “simple” relationship between observable quantities (equations)
- Predictive modelling
  - y = F(x) unknown function approximation, x: easy to measure, y: would be useful to estimate
  - The accuracy is essential, understanding is secondary
  - Examples: targeted advertisement, stock-predictions, Go-chess next best step “prediction”, sound language processing, image recognition
  - Useful in scientific research too!

$$E = mc^2$$



WHAT IS SUPERVISED LEARNING?

- Formally: estimate (approximate) a function, **y = f(x)**
  - **x** can be measured **y** is either impossible (future), very expensive/hard (e.g.: spectrum of 10 million galaxies)

- An approximate **y** can help to
  - narrow down candidates for expensive procedures
  - use approximate values for further analysis
  - make some money on the stock market

- **Learning:** building the estimated **f(x)** function using example data points: **x-y** pairs
  - Mostly: learning is the optimisation of at the parameters of f function
- **Y** is the supervising signal

TYPES OF SUPERVISED LEARNING

- Formally: estimate (approximate) a function, **y = f(x)**

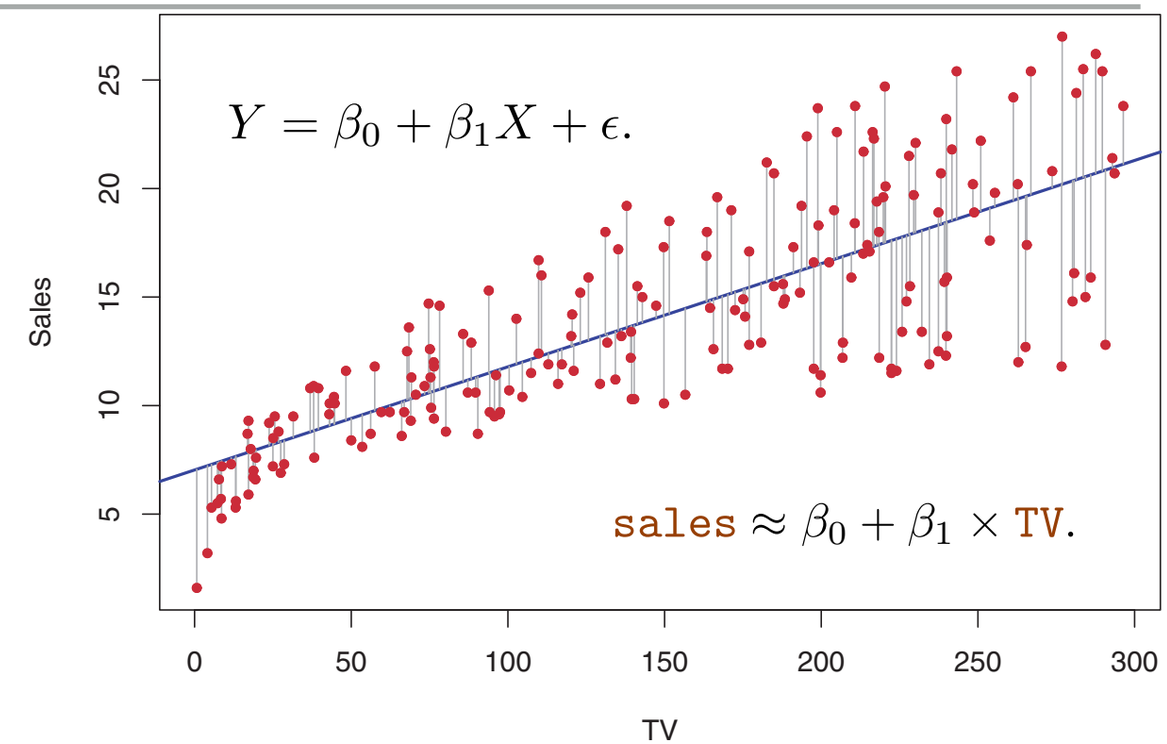
- **Y** is a continuous value: **regression**
- **Y** is discrete, but ordered: **ordinal regression/classification**
- **Y** has discrete values with no ordering: **classification**

- Regression example: stock price, redshift
- Classification example: dogs/cats on image, patient has cancer
- Ordinal regression/classification example: how many dogs are on the image

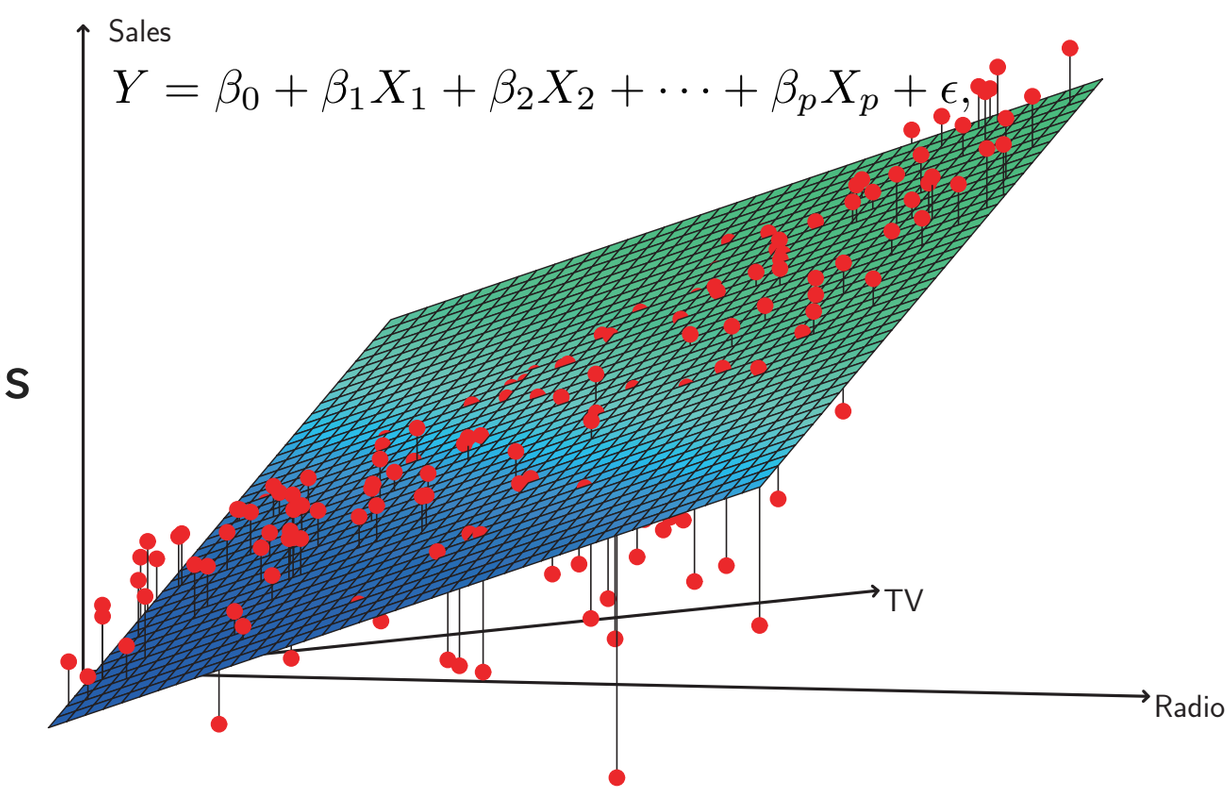
- Mostly you will predict class **probabilities**, not just classes when classifying

LINEAR REGRESSION

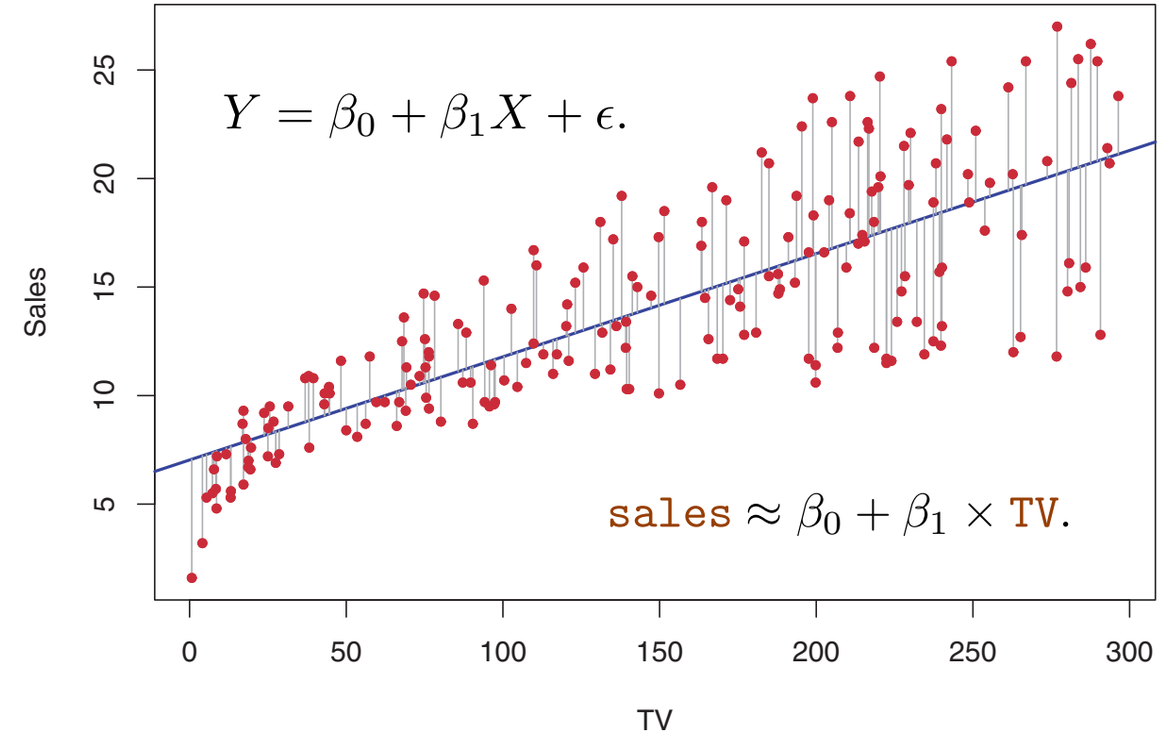
- Starting point
- Simplest parametric function
- Easy to interpret the parameters: intercept, coefficients: unit change in x makes coefficient times unit change in y
- Can be very accurate in certain problems
- Least squares
  - Insight: minimising log probability (actually the likelihood) of observations given Gaussian y distribution



$$P(\text{data} \mid \text{model}) \propto \prod_{i=0}^{N-1} \left\{ \exp\left[-\frac{1}{2} \left(\frac{y_i - y(x_i)}{\sigma} \right)^2 \right] \Delta y \right\}$$



LINEAR REGRESSION

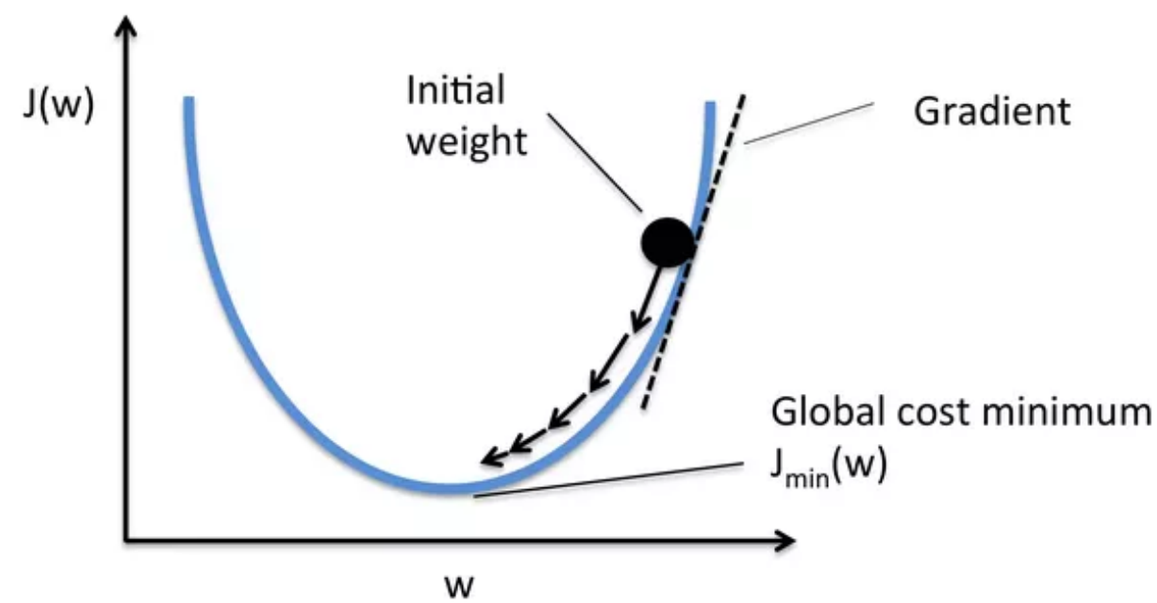


- Least squares fitting: finding the coefficients to minimise the least squared error

- Generally: we have a 'Loss' (here the least squares)

$$P(\text{data} \mid \text{model}) \propto \prod_{i=0}^{N-1} \left\{ \exp\left[-\frac{1}{2} \left(\frac{y_i - y(x_i)}{\sigma} \right)^2 \right] \Delta y \right\}$$

- We want to tune the parameters of our approximating function to minimise this Loss



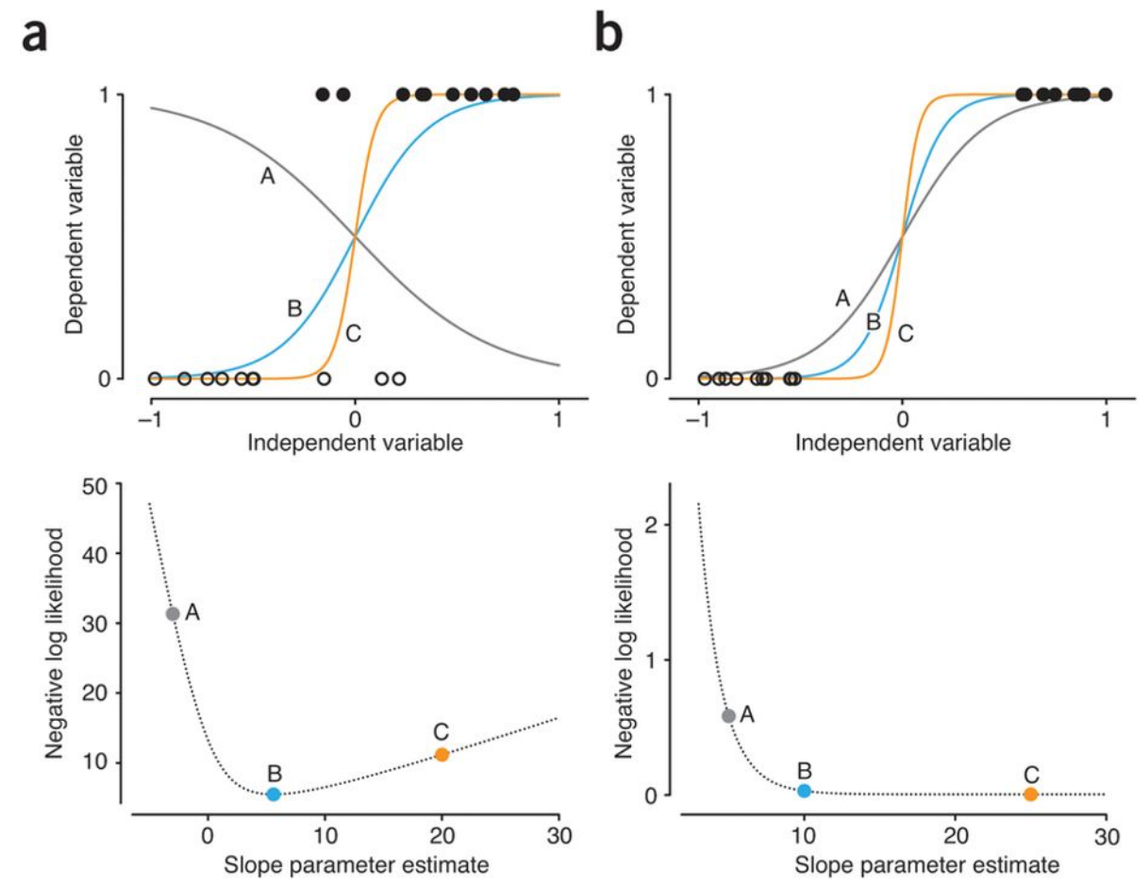
- Loss is a function of the parameters, we minimise, (optimise) the 'Loss function'

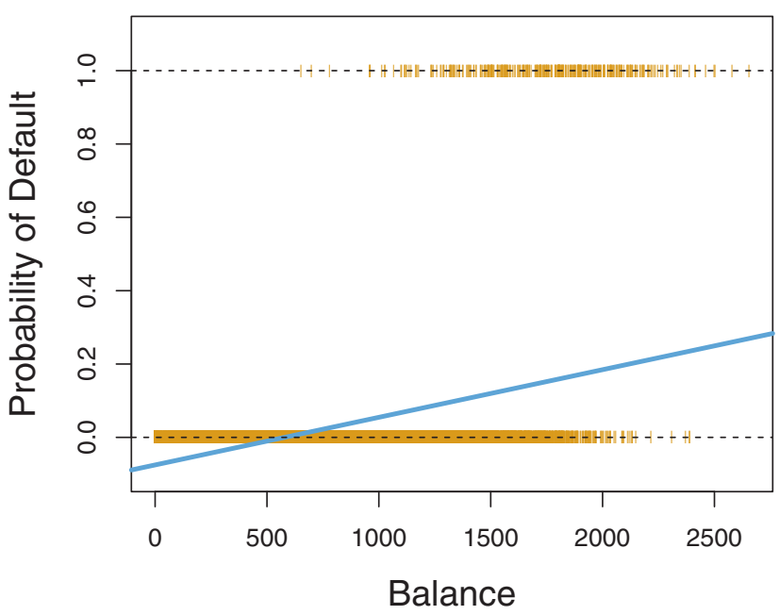
- **Learning = optimisation**

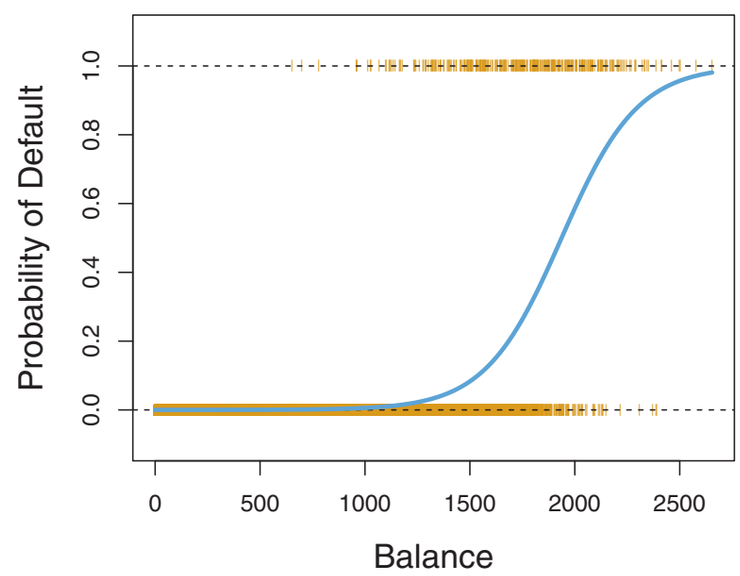
LINEAR METHODS IN CLASSIFICATION

LOGISTIC REGRESSION

- Predict continuous class probabilities (0-1) for each class, sum=1
- We could use linear regression, just it will lead to some values <0 and >1 (we could truncate…)
- But we can also transform the output of a linear model using a saturating function!







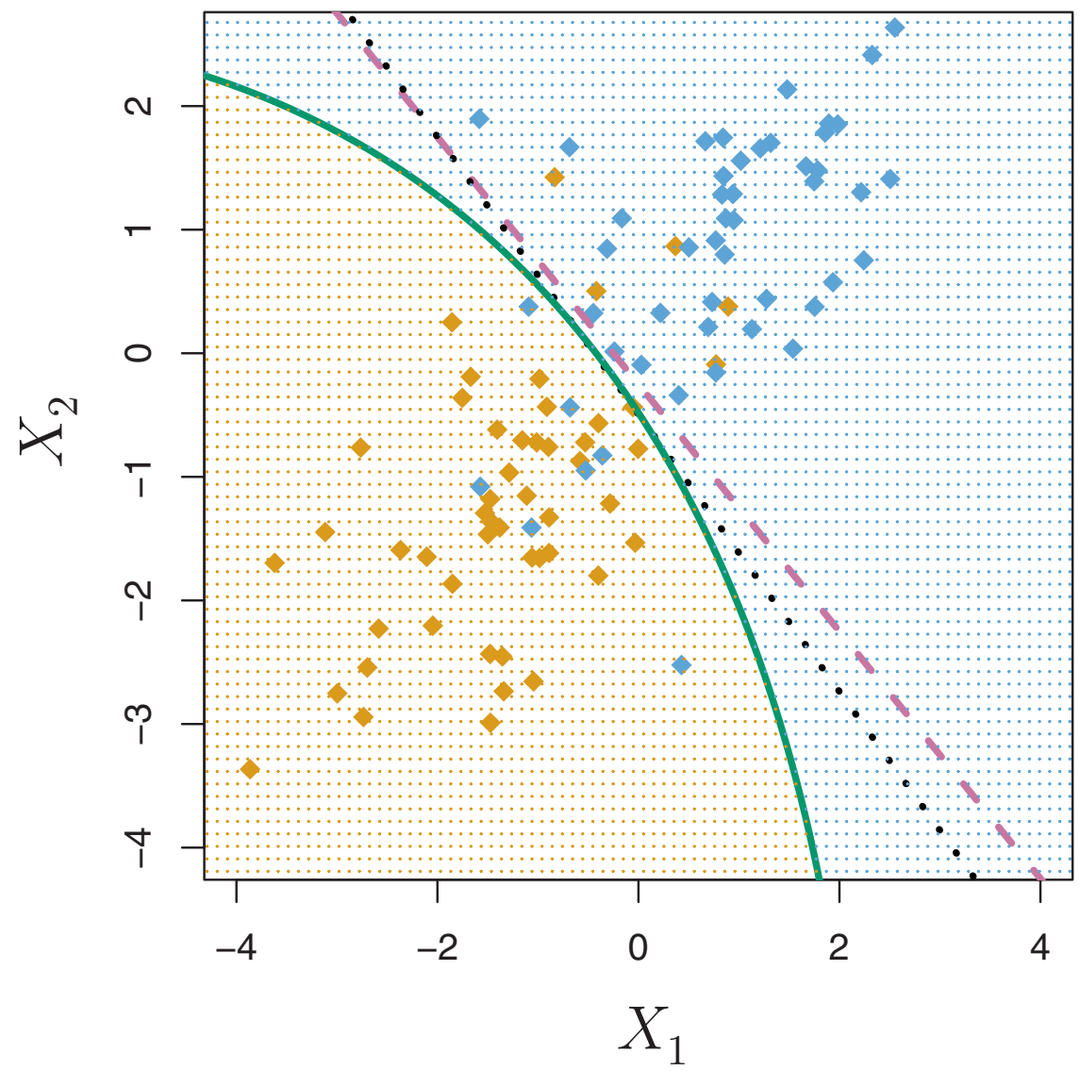
$$Logistic(y) = \frac{e^y}{1+e^y}$$

$$p(X) = \frac{e^{\beta_0+\beta_1 X}}{1+e^{\beta_0+\beta_1 X}}.$$

LINEAR SEPARATION

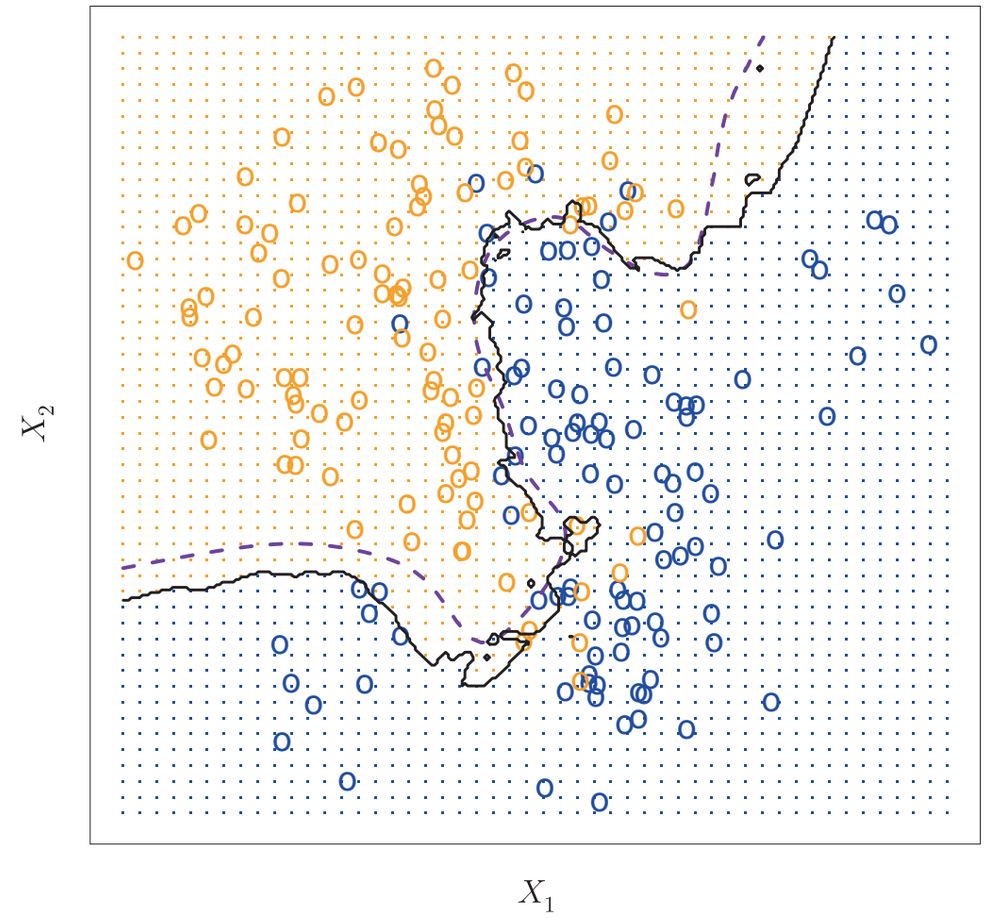
$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X.$$

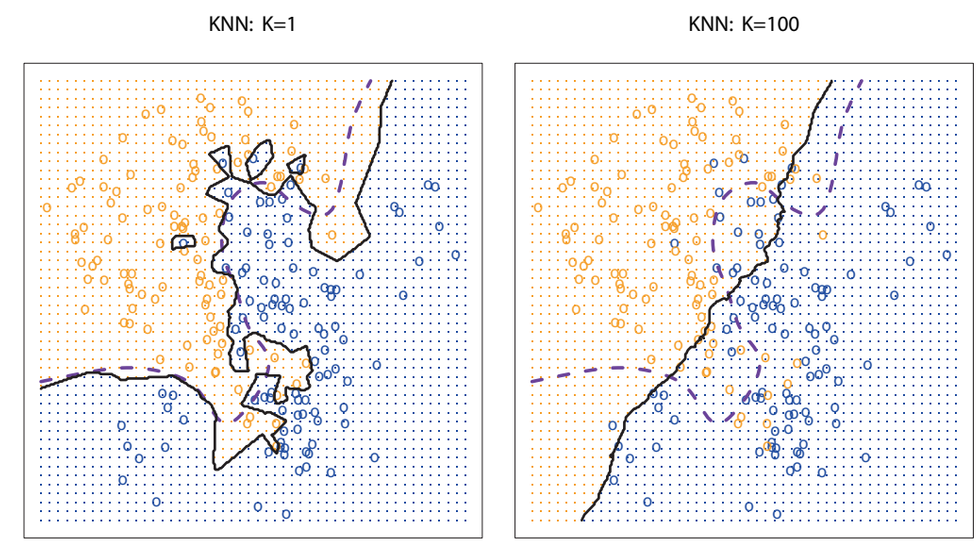
- Logistic regression is a linear model
- Classes are separated with lines/ hyperplanes
- Pick a p for classification (0.5) -> it defines the separation



K NEAREST NEIGHBOUR MODEL

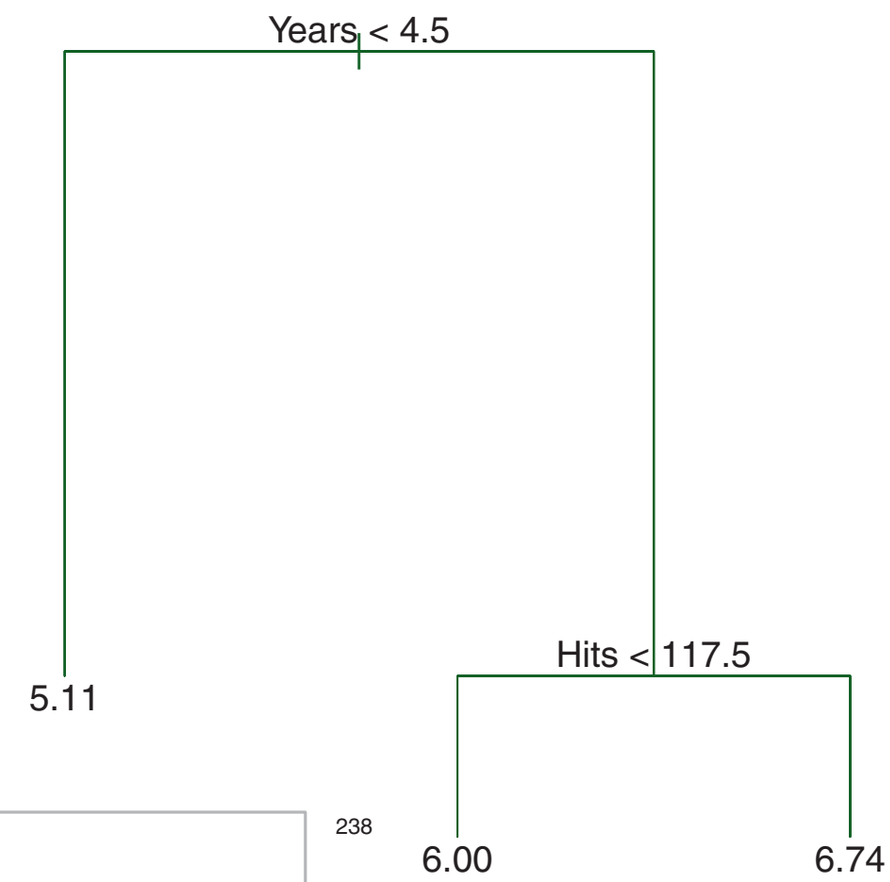
- Very simple concept for function approximation: y = f(x)
- You need to predict a new **x**, Find the closest **x** value(s) in your "training set", use the corresponding **y** for prediction!
- Simple versions:
  - classification, predict the most common **y** among neighbours
  - regression: take the average of **y** among neighbours
- Naive implementation is super slow! Efficient indexing of **x** data is needed for fast distance queries.

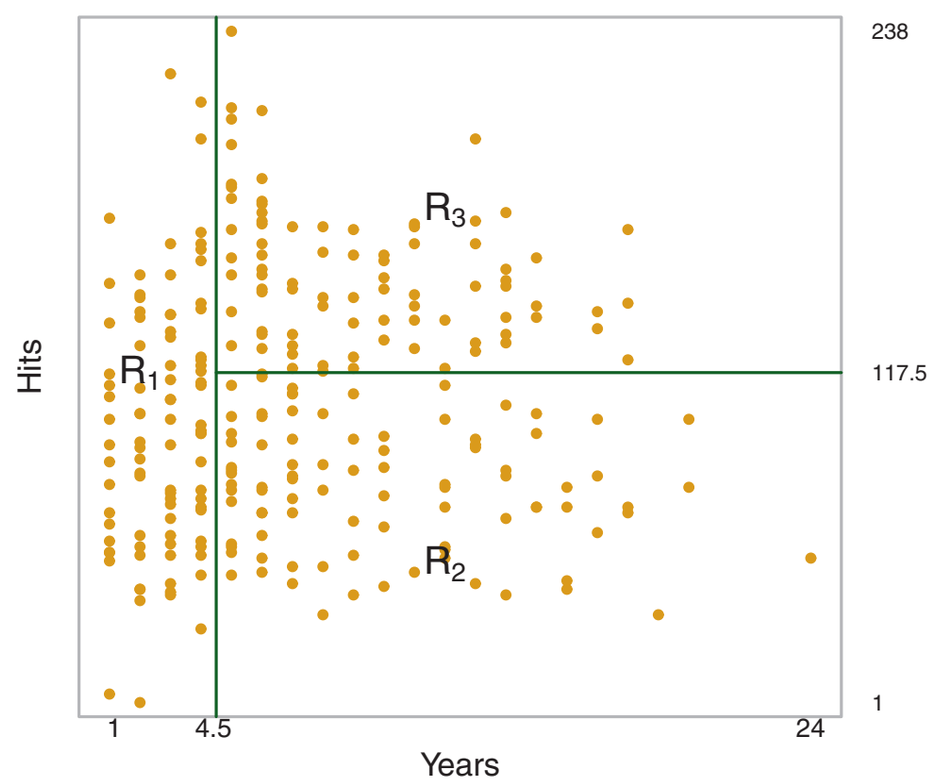




DECISION TREES

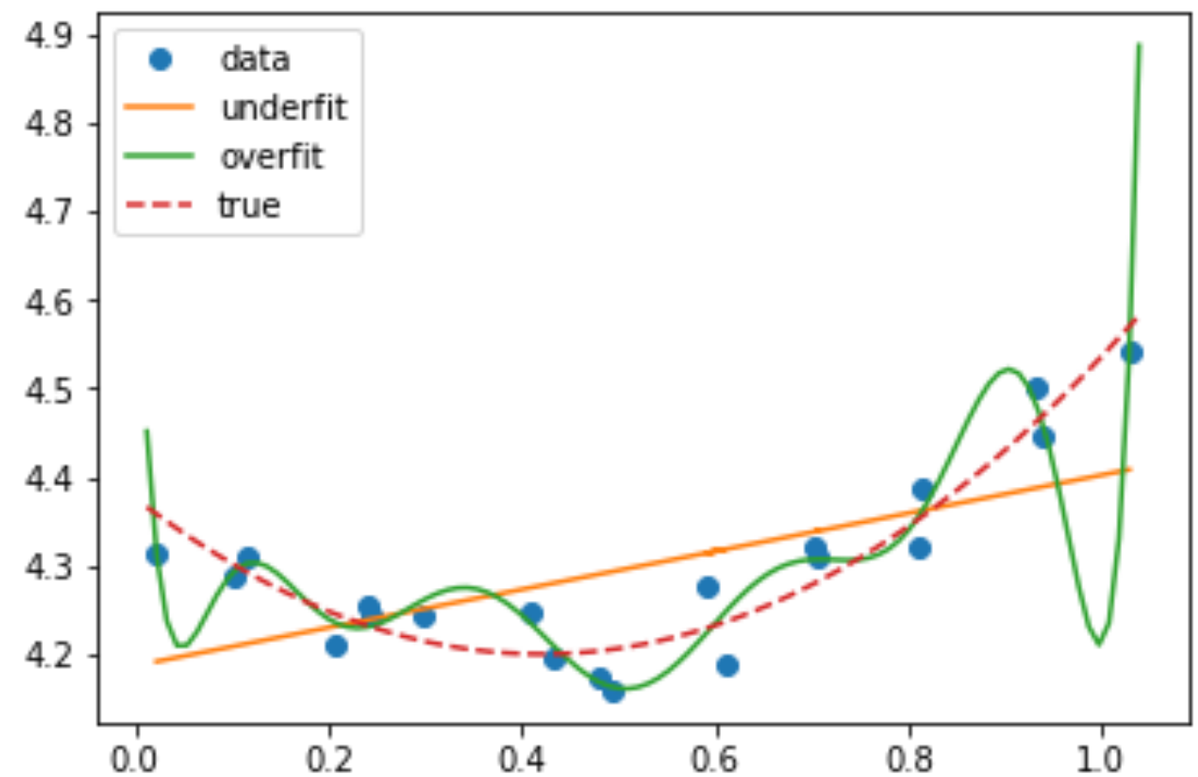
- KNN: Using the y values of the closest neighbours in a simple parameter free way
- Trees: Dividing the input space into regions with constant predicted y values
  - Divisions along the axes (single parameter)
  - nested divisions
- Super intuitive, very similar to human thinking! Easy to interpret!
- Trees ensembles (forest) are super competitive!





MODEL CAPACITY, OVERFITTING / UNDERFITTING

- There is a true P(y|x) generated by nature
- We estimate the relationship with y=f(x)
- The selection of the f function is crucial
  - Too simple: can not capture the complexity of P(y|x), suboptimal: **underfitting**
  - Just right
  - Too complex: fits the noise, not the true P(y|x), does not generalise to unseen data: **overfitting**
- Capacity of a model



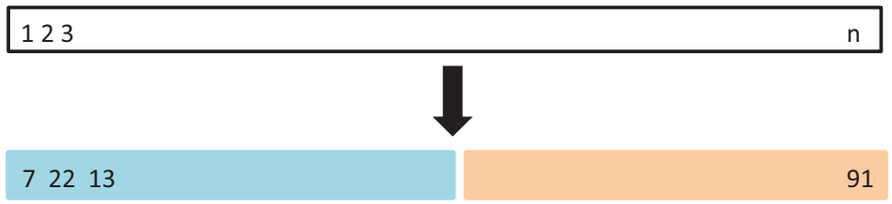
ESTIMATING THE ACCURACY OF THE MODEL

- We train the model with available data, and we will predict on unseen examples. Why do we need to estimate the accuracy of our model?
  - To know what to expect (should we risk our money on our stock market predictions)
  - To **select the most accurate model!**
- **Model selection**
    - The type of the model (knn, linear, trees, svm, nn, etc)
    - Bets combination of input variables or engineered features
- Usually we can not estimate accuracy on the training data!
  - Each model is able to learn the noise to some varying extent. How much worse it will be on unseen data? We cannot really tell.

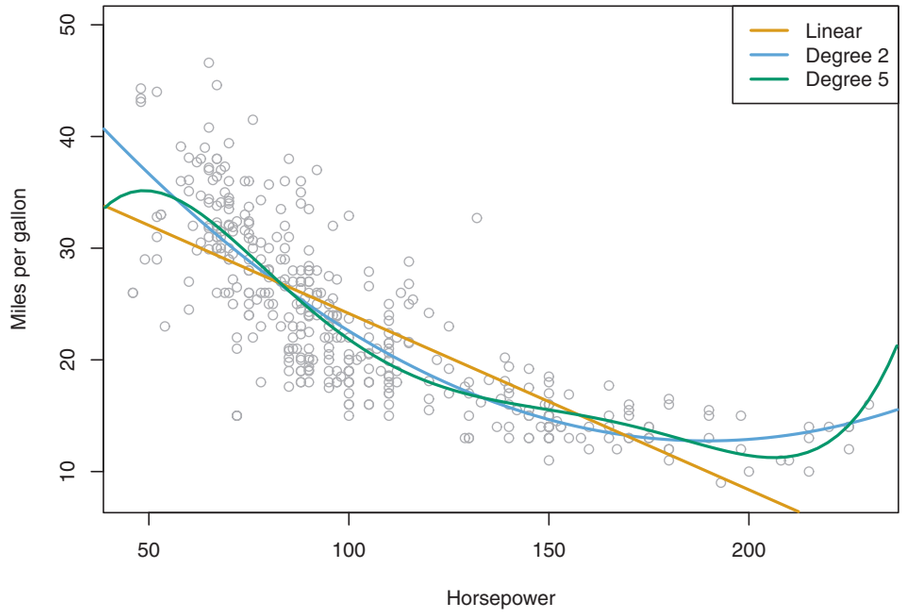
SUPERVISED LEARNING IS NOT SIMPLY OPTIMISATION!

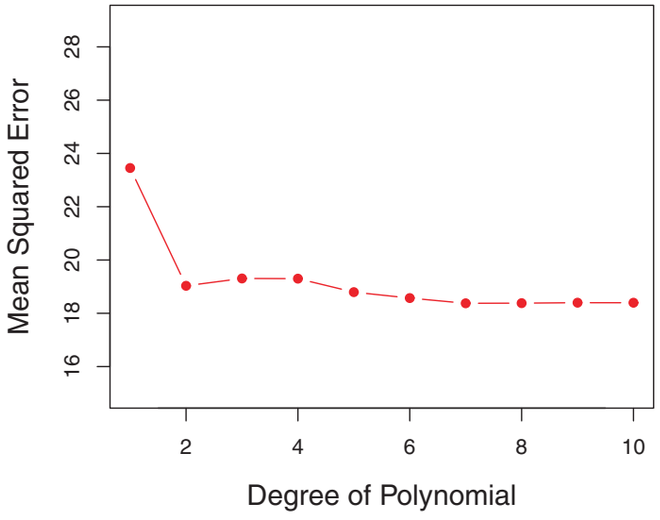
- We need a function which approximates **y = f(x)** on **unseen new examples** not on the data we already have!
  - A database is a good solution to minimise the loss function on the examples we have
  - Accuracy on the set used for training/optimisation is meaningless! (It may help debugging of the model, though)
- The problem is not optimising the loss function on the data we have, It is optimising the loss function on data we do not have. Obviously the problem is **not a well-defined optimisation**.
- Estimations are always evaluated with **held-out data!**
  - Test data must come from the same circumstances as we intend to use the model for the useful predictions (the same x-ray manufacturer, the same experimental kit for gene expression measurement, the same telescope, the same method of collecting user/company information, etc). Otherwise the accuracy of predictions might be very different.

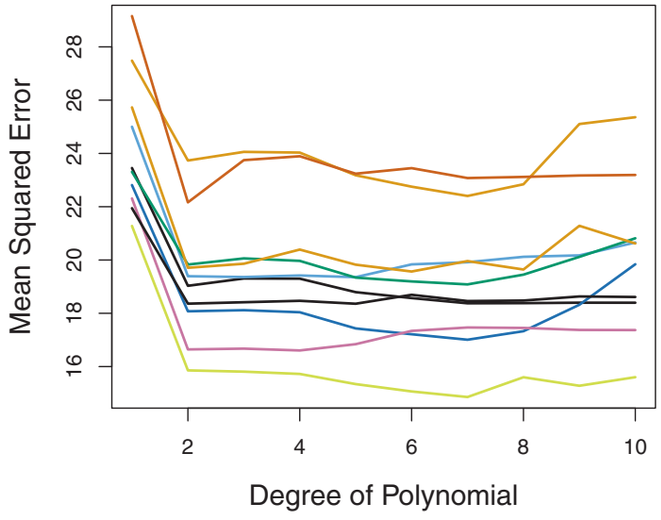
TRAINING – VALIDATION SPLIT



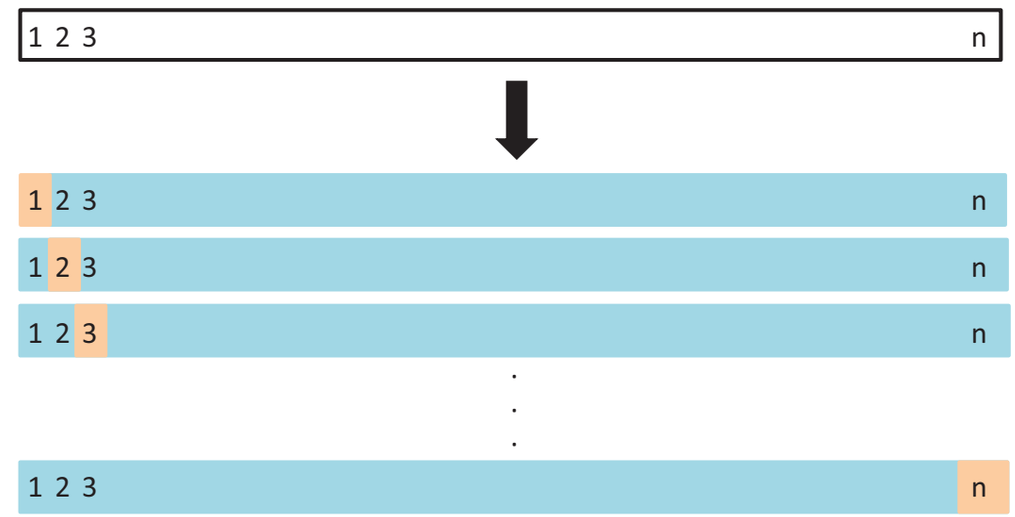
- Cut the training data into 2 parts, training and testing
- Not ideal:
  - Not using all data for training -> model will be not as accurate as on the whole dataset (we overestimate the error)
  - Not using all data for testing: our estimate will not the most accurate
- Typically used when datasets are huge, and training is very expensive: this is the standard in image recognition.
- Note, that in small sets the accuracy may be widely varying among different splits.



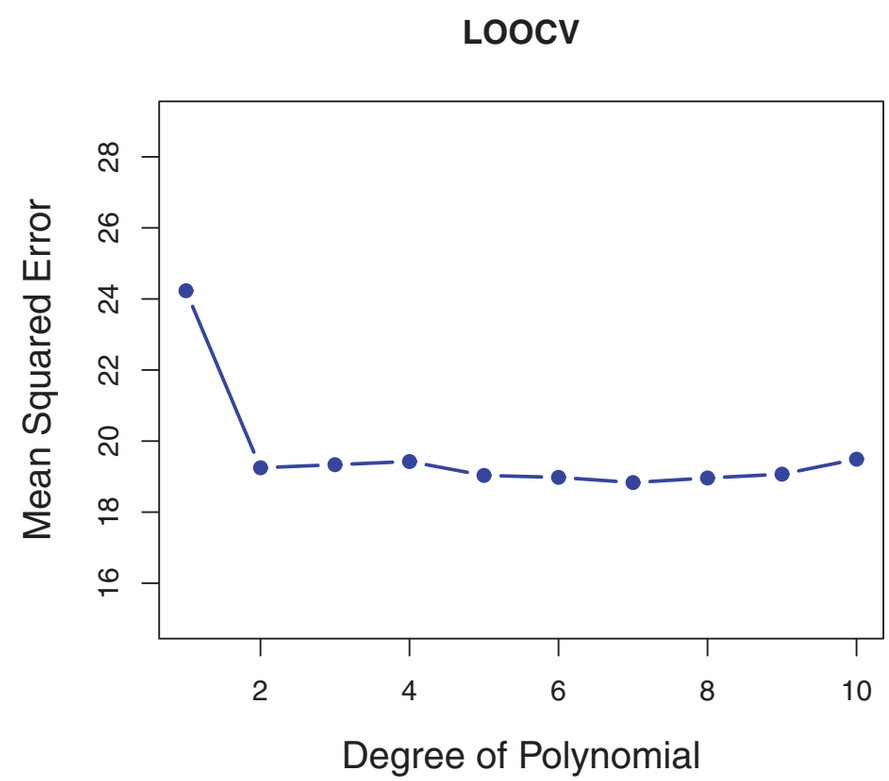




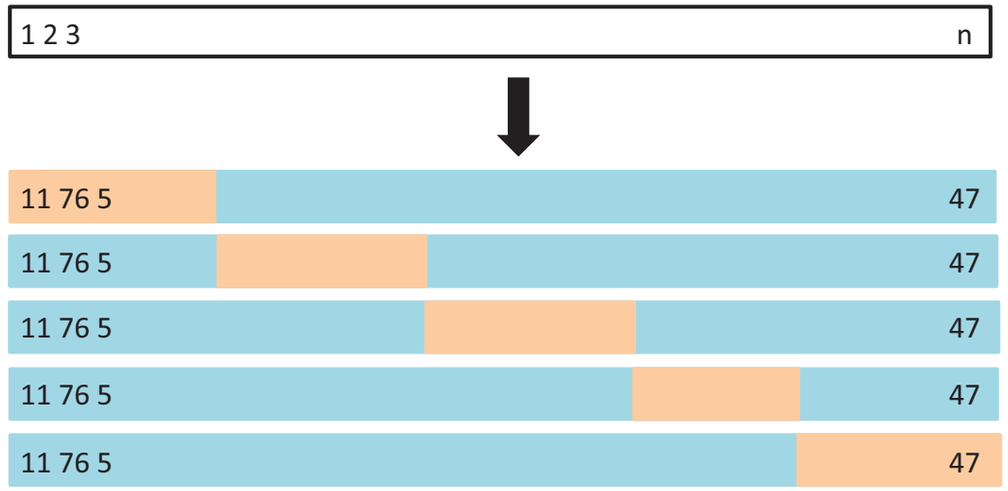
LEAVE ONE OUT CROSS VALIDATION



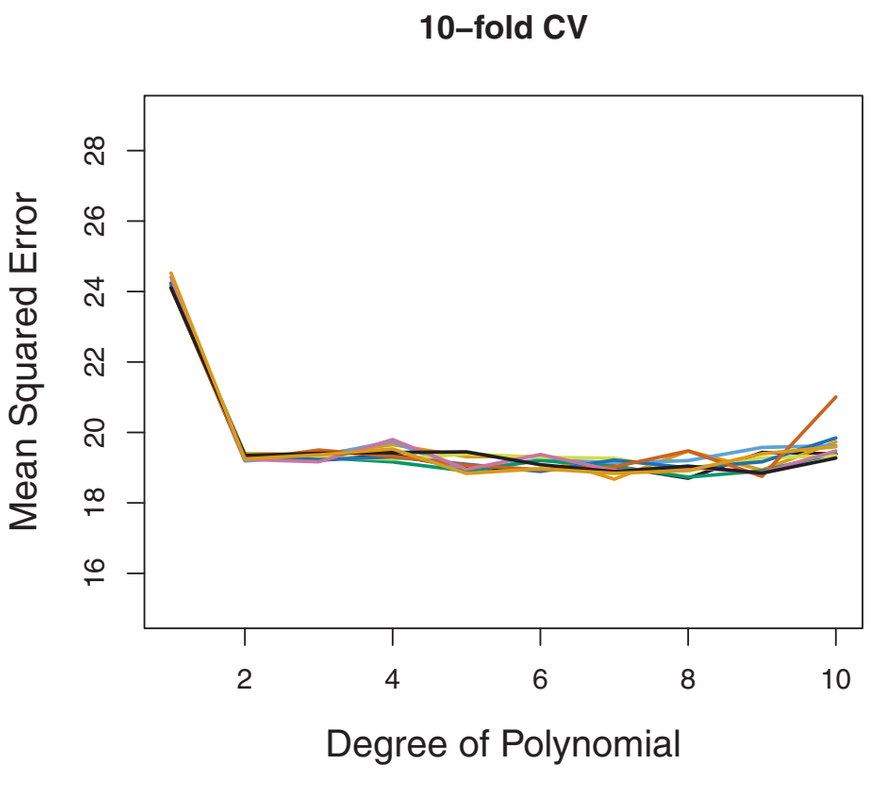
- With N data points, cut the training data into 2 parts, training and testing, N times. Each point is a test on its own once.
- The accuracy is estimated as the mean accuracy on each example
- Almost each data point is used in training: the model will be as good as possible
- Each data point is used for estimation, estimation of accuracy will be good.
- It is practically impossible on any normal dataset and complex model.
  - Tens or hundreds of thousands of data points, the model has to be fit tens or hundreds of thousands times, complex models could take minutes or hours to fit
- Note: there is no random process!



K-FOLD CROSS VALIDATION

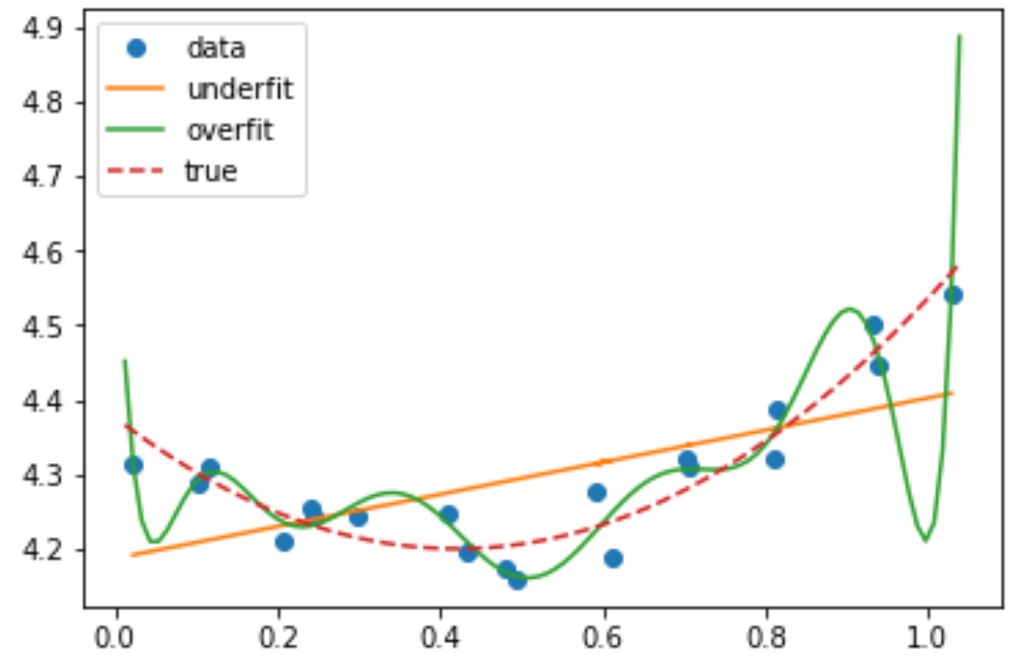


- With N data points, cut the training data into K parts, use K-1 for training and 1 for testing, K times. Each part is a test on its own once.
- K is usually 5,10 (rarely 3).
- The accuracy is estimated as the mean accuracy on each set.
- 80-90% of the data points are used in training: the model will be close to be as good as possible
- Each data point is used for estimation, estimation of accuracy will be good.
- Usually can be done for large sets, model needs to be trained only 5-10 times. Only exception is image recognition, because then it is too expensive to even train 5 times (a week vs a month).
- Note: there is a random process of splitting, but variation is not as wild as in a train-validation split because the same data points are used.
- **THE STANDARD**

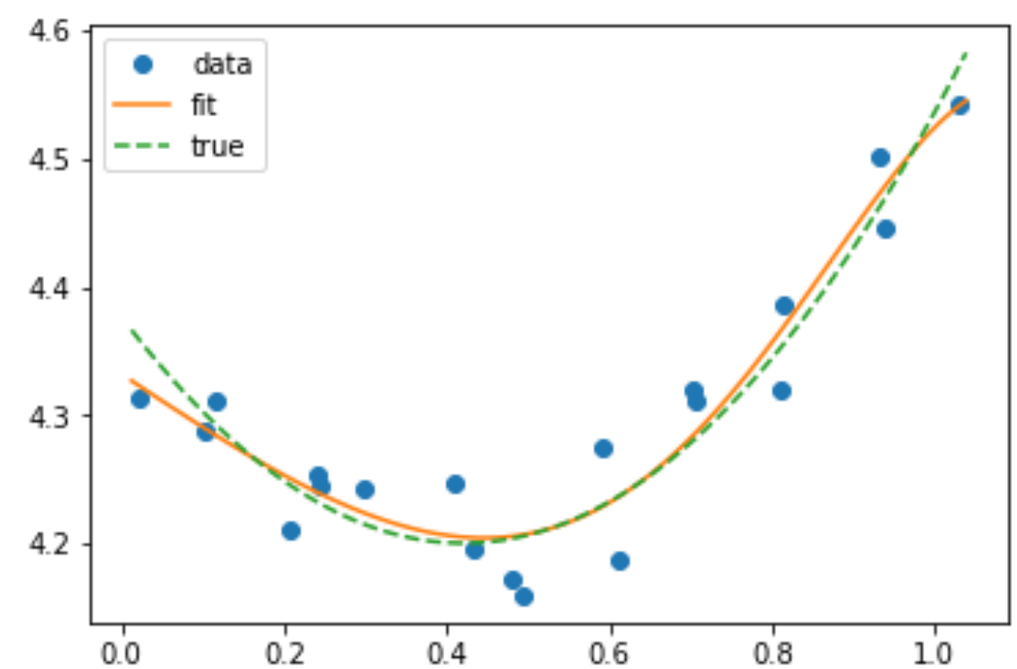


REGULARISATION

- Subset selection (training error is larger!)
- Shrinkage (training error is larger!)



- Optimisation: reduce error/loss of training data
- Regularisation: reduce error/loss of unseen data, while increasing training error
  - Improving generalisation of a model: simpler, smoother models
  - Most often meant inside a function family, (but the structure of function can be though of as a regularisation)



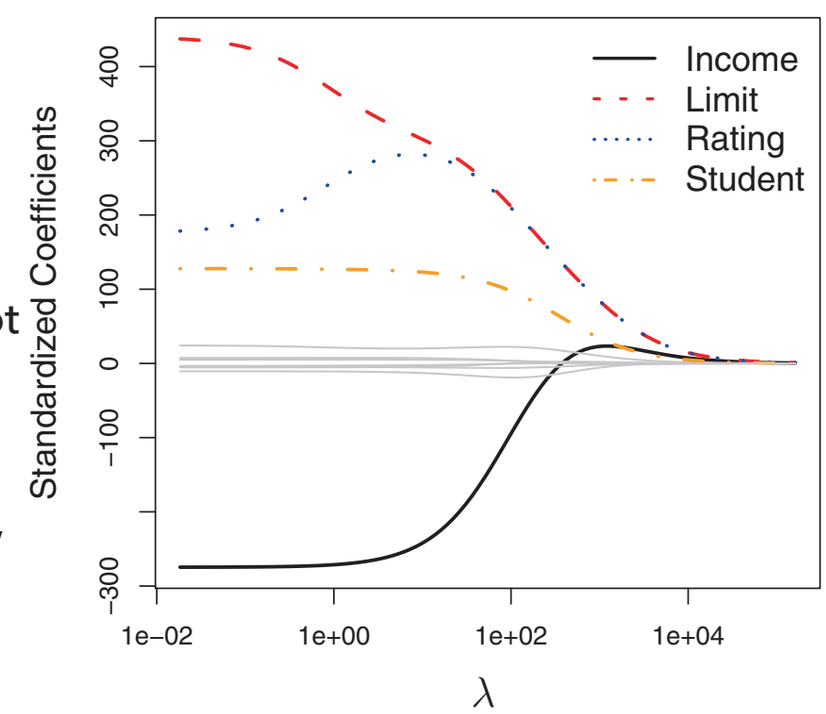
- Controlling the capacity of functions to match the problem: large capacity model + regularisation

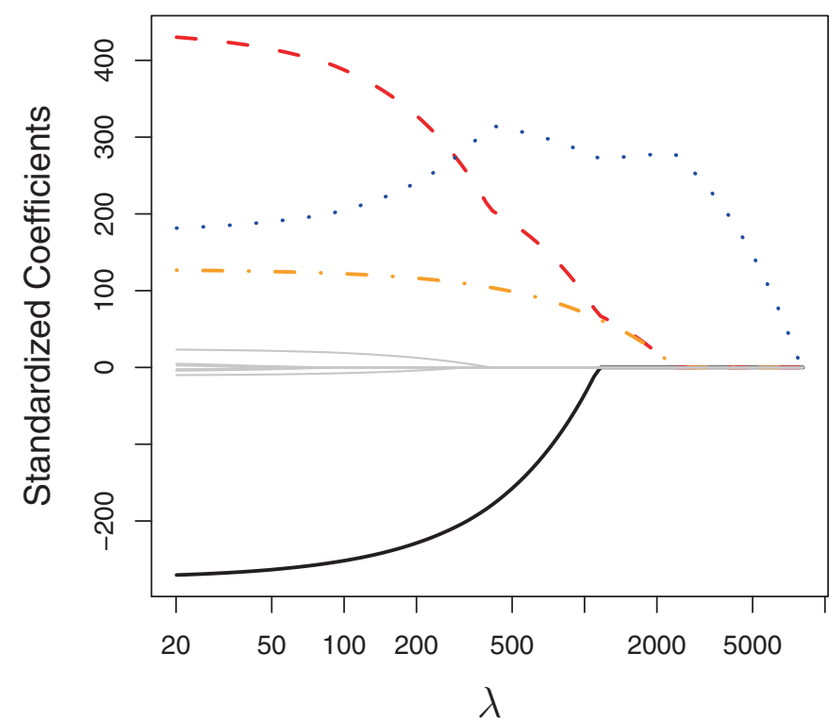
SHRINKAGE (RIDGE, LASSO)

- Subset selection: a simple model has fewer predictive variables
- Shrinkage: a simpler model has **smaller** coefficients, instead of cancelling large coefficients
- Ridge-regression, L2 penalty (Weight-decay) Lasso-regression, L1 penalty (sparse: often results in 0 coefficients), Elastic-net: both
- They increase the training error! (but hope to increase generalisation by creating a simpler model)
- Essential for underdetermined problems! (SVD would not fail but not ideal)
- Note: The scale of predictive variables did not matter before! Now it does!

$$\sum_{i=1}^{n}\left(y_i-\beta_0-\sum_{j=1}^{p}\beta_j x_{ij}\right)^2+\lambda\sum_{j=1}^{p}\beta_j^2=\mathrm{RSS}+\lambda\sum_{j=1}^{p}\beta_j^2,$$

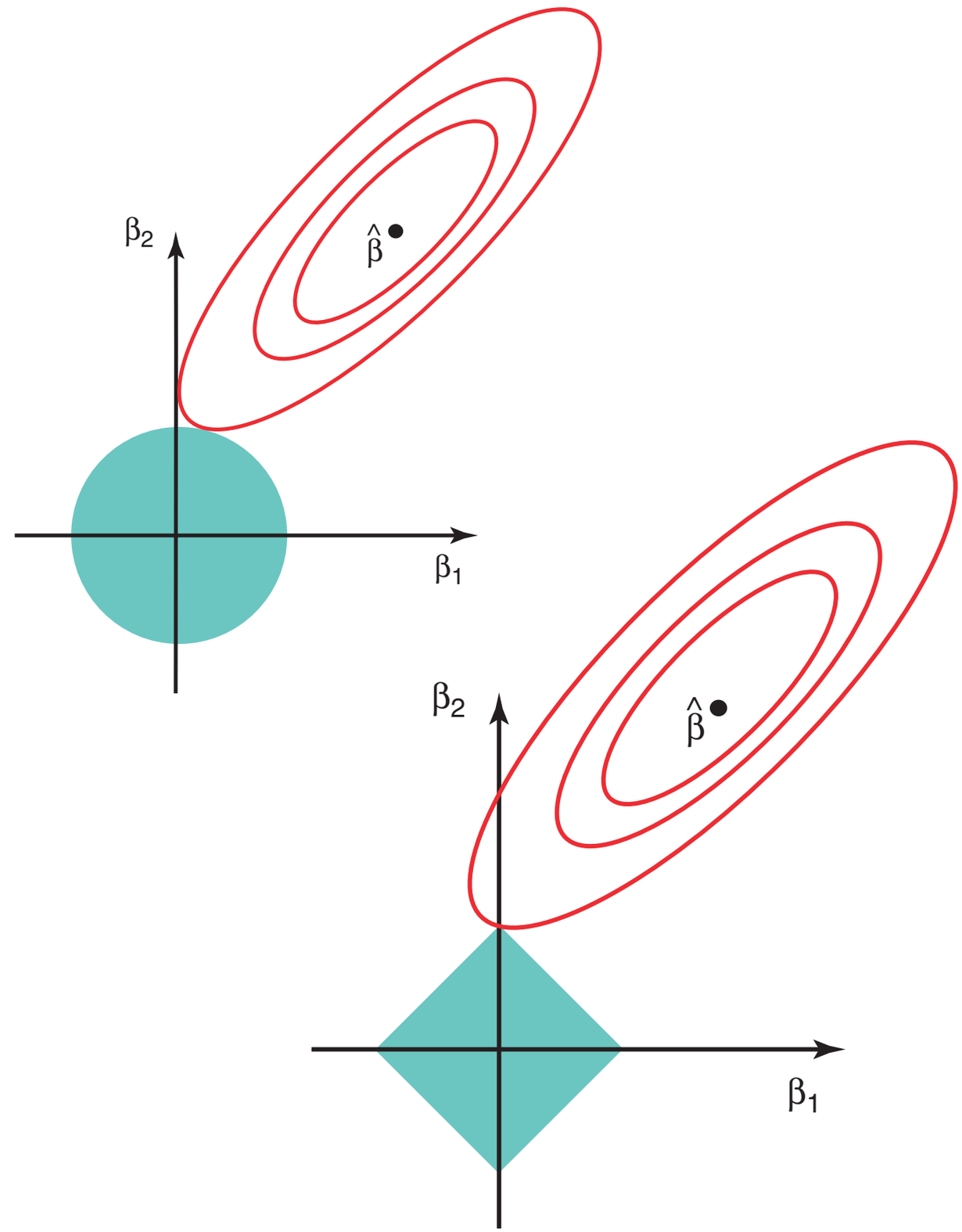
$$\sum_{i=1}^{n}\left(y_i-\beta_0-\sum_{j=1}^{p}\beta_j x_{ij}\right)^2+\lambda\sum_{j=1}^{p}|\beta_j|=\mathrm{RSS}+\lambda\sum_{j=1}^{p}|\beta_j|.$$





SHRINKAGE (RIDGE, LASSO)

- Subset selection: a simple model has fewer predictive variables
- Shrinkage: a simpler model has **smaller** coefficients, instead of cancelling large coefficients
- Ridge-regression, L2 penalty
  - Weight-decay
- Lasso-regression, L1 penalty
  - Often results in 0 coefficients
- Essential for underdetermined problems! (SVD would not fail but not ideal)
- Enables us to train models on more variables than data points! Still quite controversial…
  - https://www.nature.com/articles/d41586-018-06956-3



REFERENCES

- James, G., Witten, D., Hastie, T. and Tibshirani, R., 2013. **An introduction to statistical learning** (Vol. 112, p. 18). New York: springer.
  - http://www-bcf.usc.edu/~gareth/ISL/