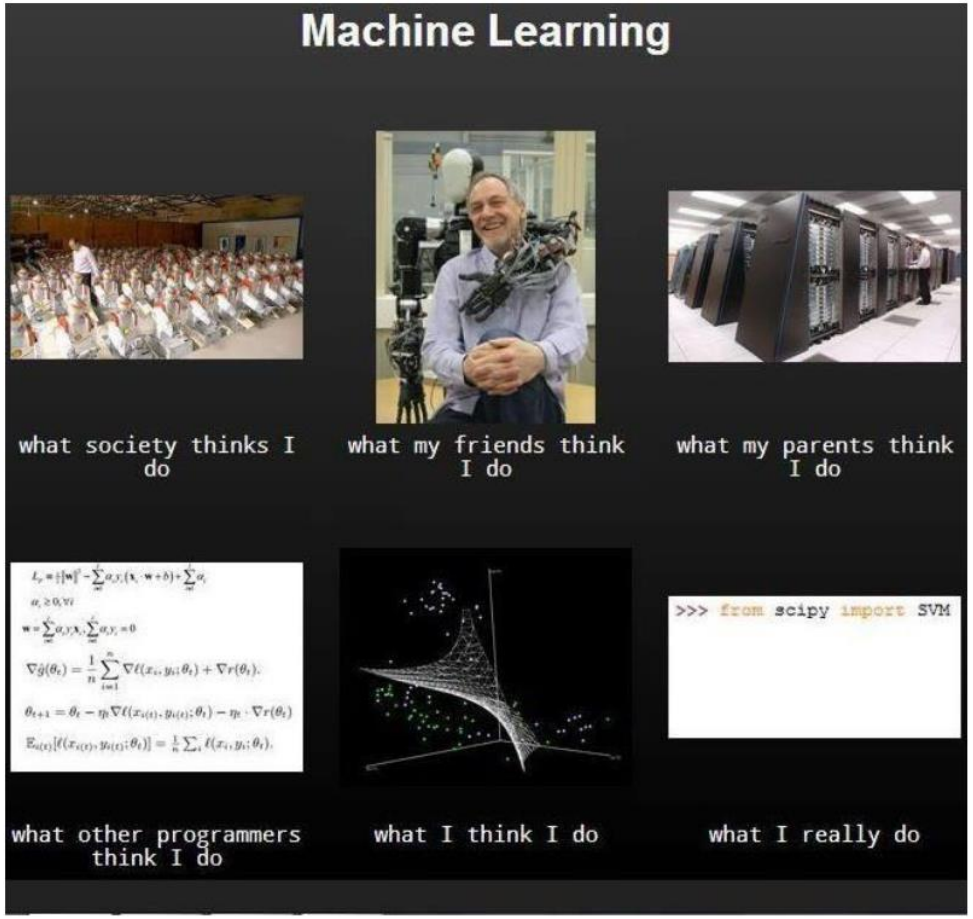
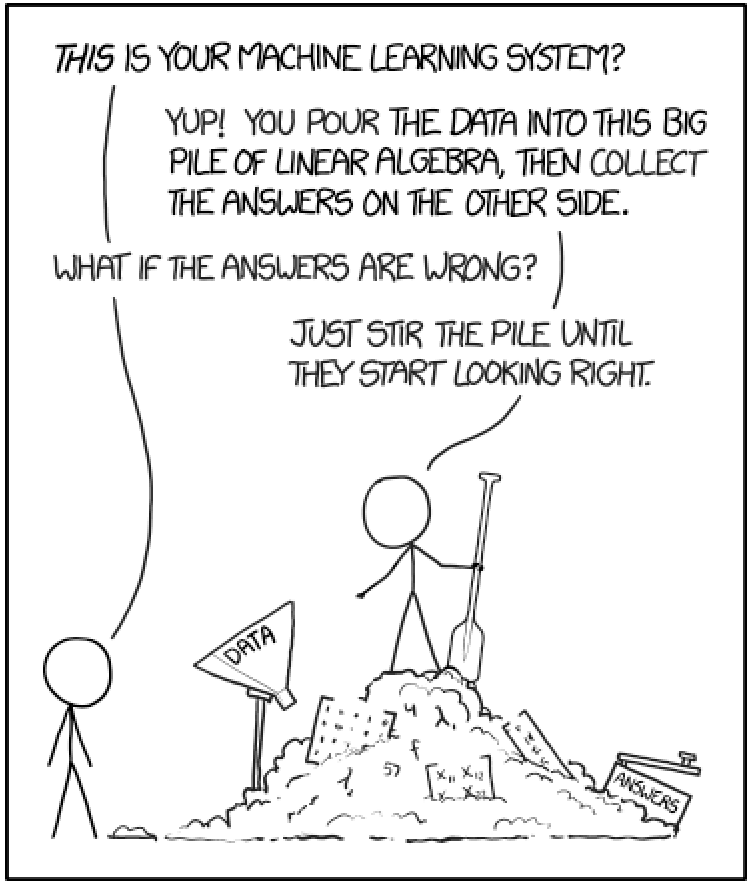
CAB420: Machine Learning Basics

WITH PICTURES AND VIDEOS

What is Machine Learning?

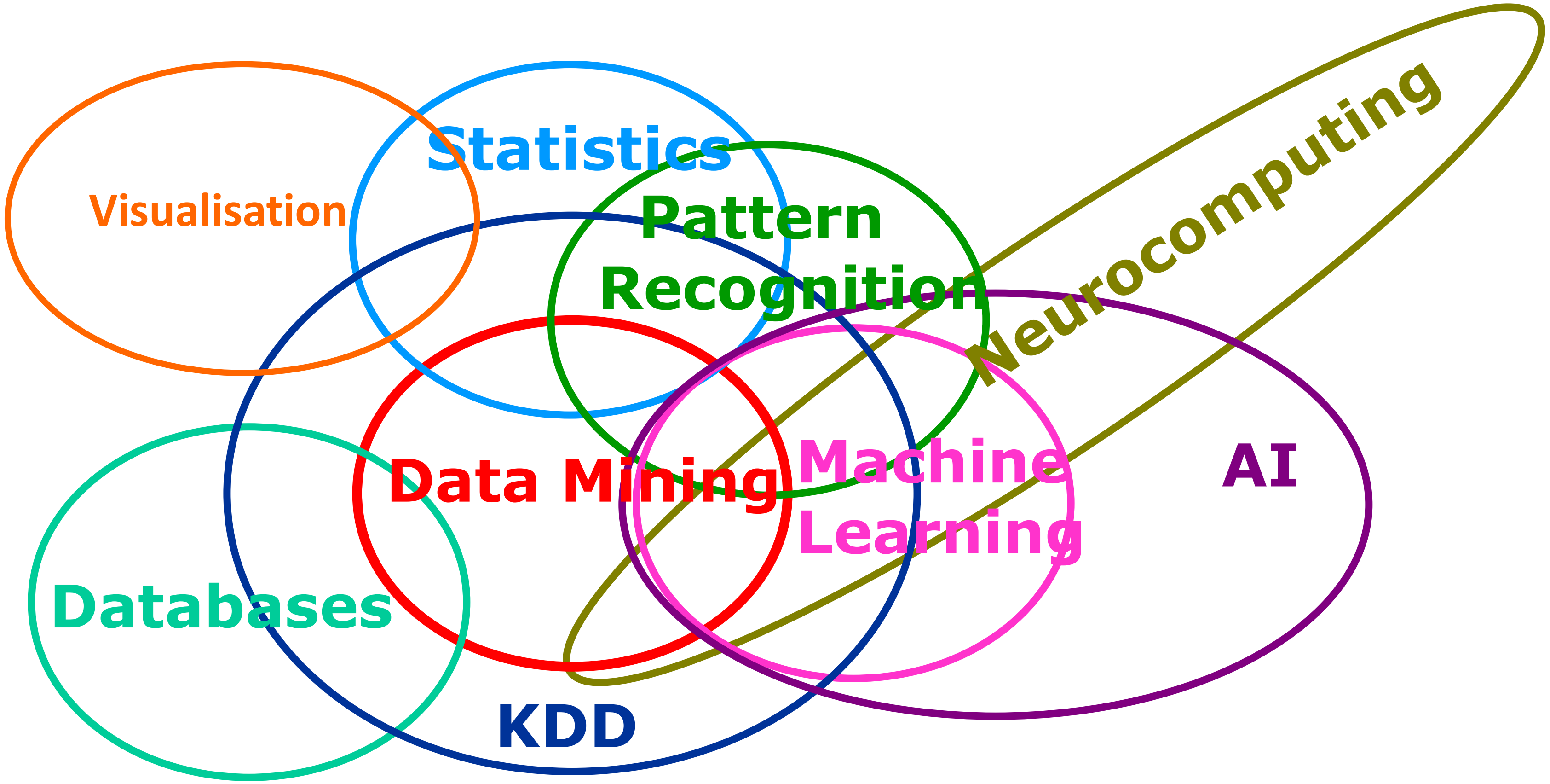


What is Machine Learning?

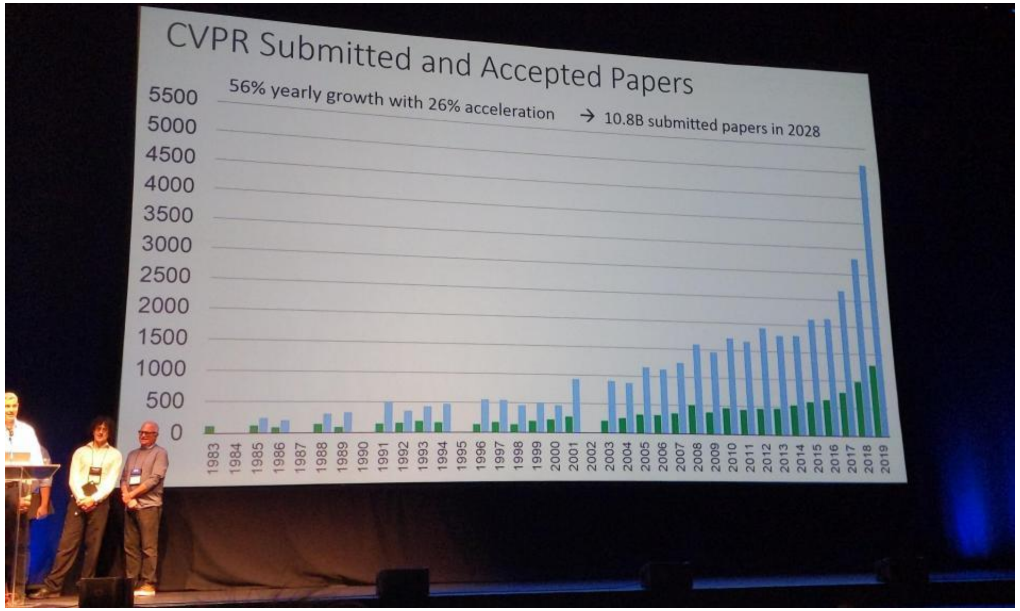


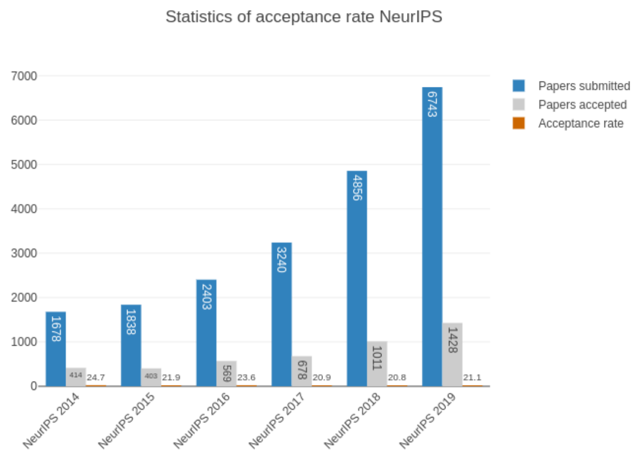
From XKCD.

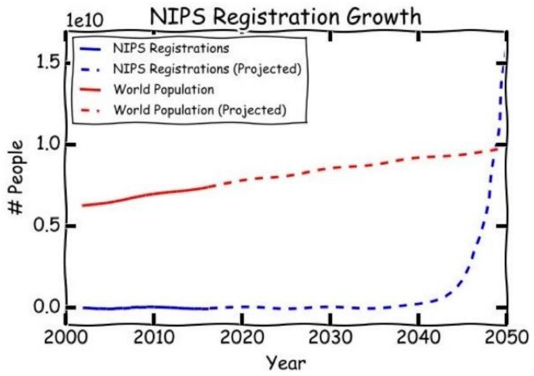
A Multidisciplinary Field



A Growing Field

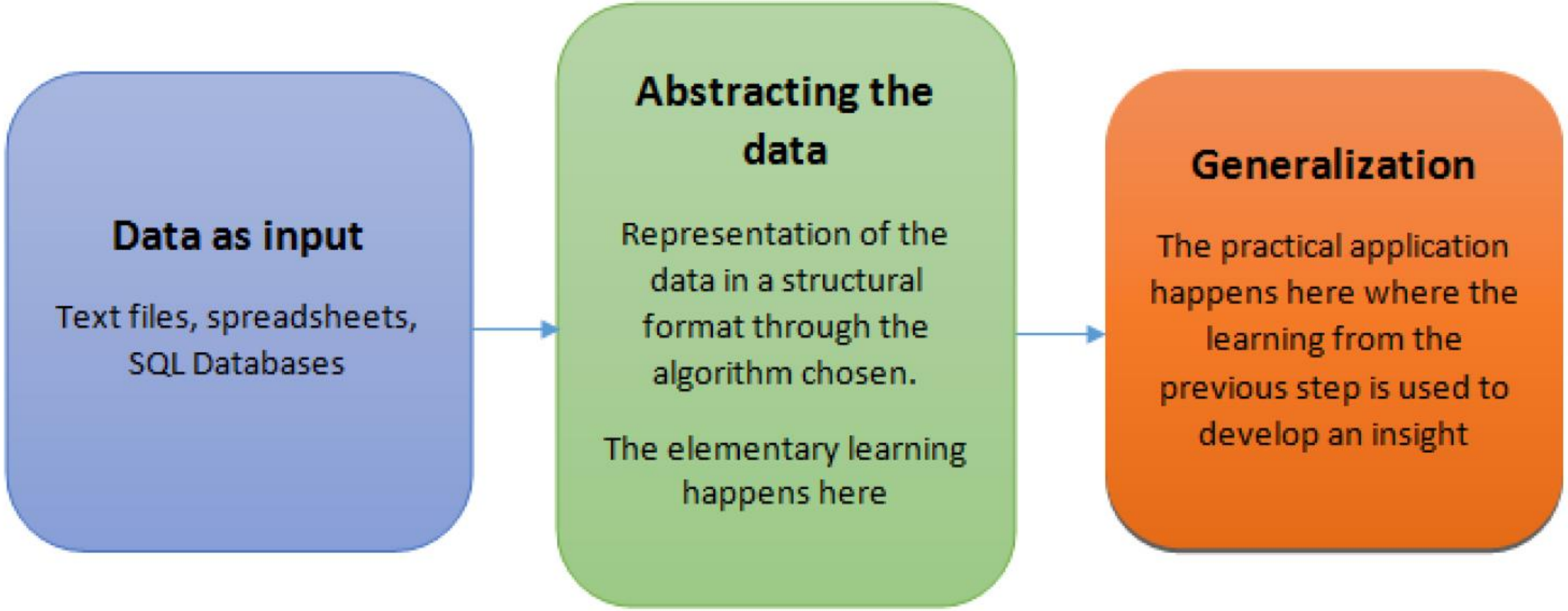




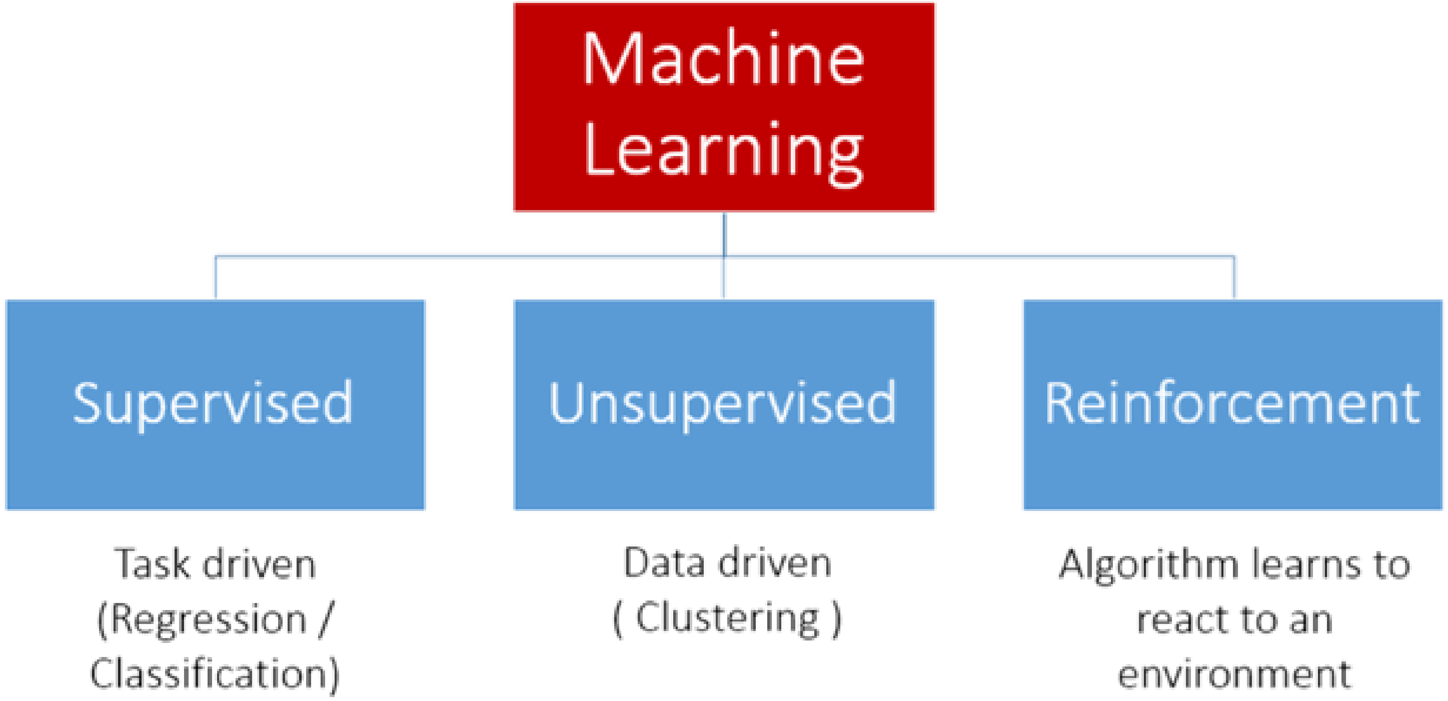


How do we teach machines?

Broadly, there are three steps



Types of Machine Learning



An Example: Regression

Regress from image features

- Size, texture, shape

to a person count

An Example: Clustering

Group pedestrians by the way that they move

- Using information extracted that describes movement

An Example: Supervised Learning

Tracking people with supervised models to:

- Detect people
- Learn how people move (path prediction)

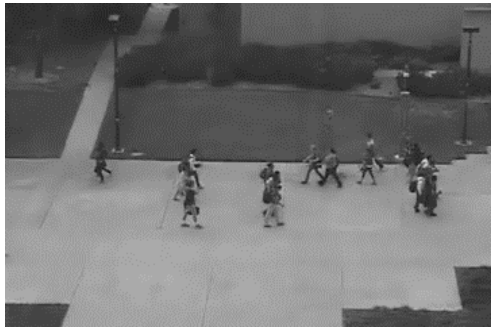
Optimisation task

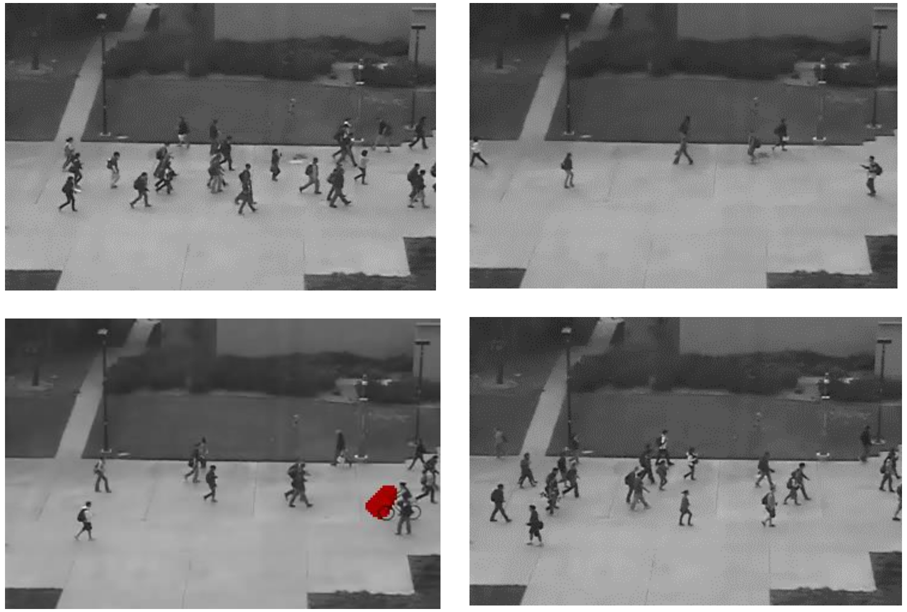
- Assignment of detections to tracks
- Hungarian Algorithm

An Example: Unsupervised Learning

Detect abnormal behaviours

- Learn from all data, data points that are rare, are abnormal

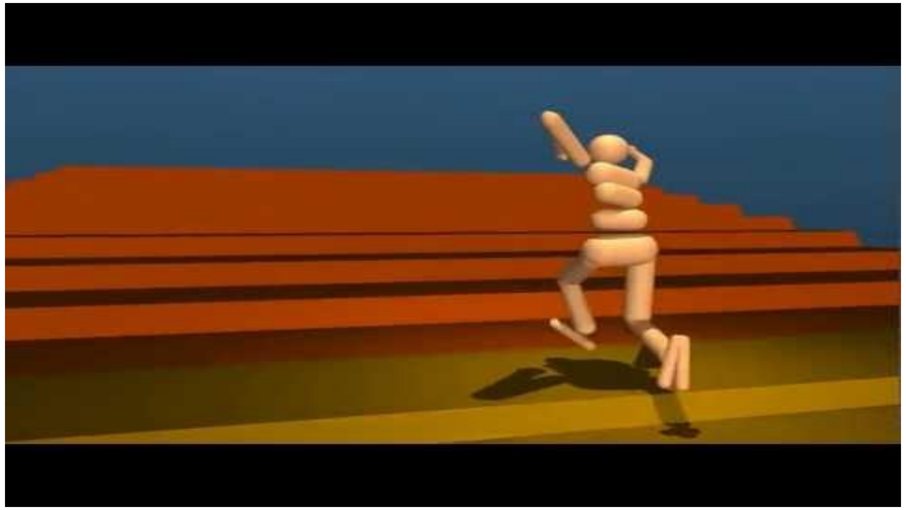




An Example: Reinforcement Learning

Learning to walk and run

- Learns from experience
- We're not covering reinforcement learning in this subject



Video from Google DeepMind

Types of Machine Learning

- Supervised
  - We have labelled data
    - For each example, we know what the answer should be

- Unsupervised
  - No labelling (though we may have prior knowledge)
    - Knowledge discovery

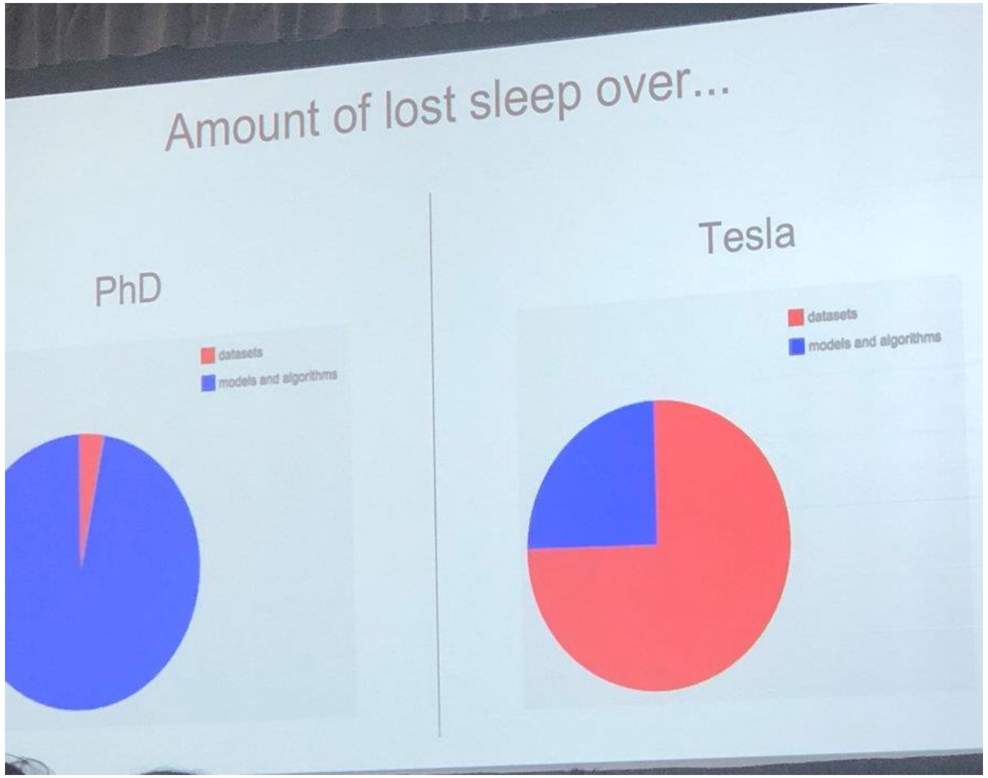
- Semi-Supervised
  - Only some (usually a small amount) of our data is labelled
    - Half-way between supervised and unsupervised
    - Learn from both labelled and unlabelled data

- Reinforcement
  - Learn through interactions with the environment
  - Not covered in CAB420

Data is Important

- In research we care more about the algorithm
- In the real world, a well prepared dataset is often more important

Slide from a presentation given by Andrey Karpathy, Director of AI at Tesla



Garbage In -> Garbage Out

- Machine learning needs data
  - But that data also needs to be collected properly

- Poor data will lead to poor models
  - No amount of fancy modelling will compensate for errors in annotation, or a poorly collected dataset

Bias and Imbalance in Data

- Models can only learn from what we provide them
  - If we provide biased data, models will learn those same biases
  - Becoming an increasing problem as we place more trust in machine learning to make decisions

- Google Photos example
  - Classified African Americans as Gorillas
    - https://www.forbes.com/sites/mzhang/2015/07/01/google-photos-tags-two-african-americans-as-gorillas-through-facial-recognition-software/#1834204d713d

- Amazon Recruitment tool example
  - Biased against women
  - https://www.theguardian.com/technology/2018/oct/10/amazon-hiring-ai-gender-bias-recruiting-engine

Data splitting for machine learning

Data splitting for machine learning

- Training
  - Data that we use to train the model
- Validation
  - Data that we use to tune model parameters. Separate to the training data.
- Testing
  - Unseen data. Emulates the “real-world” data we want to apply our model to.
- Some fields may switch the role of validation and testing

Holdout Validation

- Data is “held out” for validation and/or testing

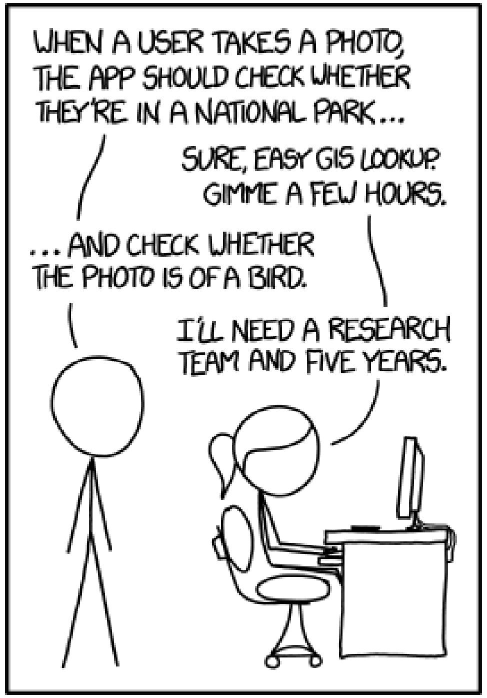
- This is usually the ideal approach
  - Training, validation and testing are all totally separate
  - Requires a large enough amount of data to be able to split it

What if we don’t have enough data?

- Cross-Fold Validation
  - Don’t create a separate validation dataset
  - Dynamically split the training dataset into training and validation (say 80/20)
  - Repeat with 5 “folds”, so that each 20% ends up being in the validation set
  - The best model on average over all folds is the one we choose

What if we don’t have enough data?

- And how much data is enough?
- Machine Learning may not be possible if we don’t have enough data
  - Our data needs to contain whatever patterns, relationship, etc, we seek to model
    - For some problems, this may be a couple of hundred samples, for others it may be millions
- Extrapolating beyond our data can be problematic
  - Needs lots of data to build a good model to allow for extrapolation
  - Hard to extrapolate from very few samples



IN CS, IT CAN BE HARD TO EXPLAIN THE DIFFERENCE BETWEEN THE EASY AND THE VIRTUALLY IMPOSSIBLE.

Cartoon from XKCD

The Curse of Dimensionality

- Datasets will often have
  - Lots of dimensions (i.e. lots of columns, observed variables)
  - Not that many samples (i.e. few rows)

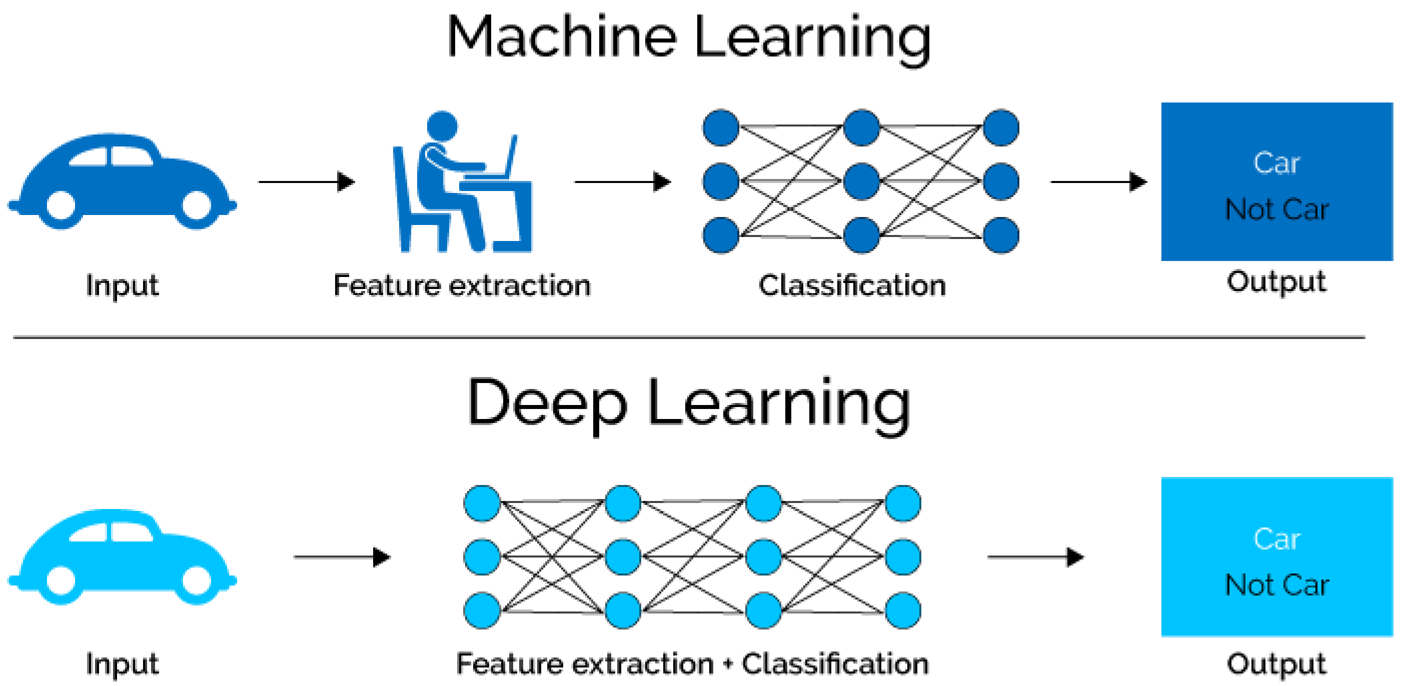
- Using all dimensions in this case may not be possible
  - Too many variables and too few samples will lead to overfitting
  - Feature space becomes sparse
    - All points are far away from each other in the N-dimensional space

- For every variable you add, you need more data
  - How much data depends on the type of classifier or learning algorithm being used.
  - Keep your models simple

CAB420: Traditional ML vs Deep ML

THE SAME, BUT DIFFERENT

Traditional vs Deep



“Traditional” Machine Learning

- Basically, this just refers to a time before deep learning
  - About 2012/13 and earlier

- Common Pipeline:
  - Pre-processing
    - Resizing, filtering, noise reduction
  - Feature extraction
    - MFCC (audio), HOG (image/video), Bag-of-Words (text)
    - Informed by the task
      - Do need to recognise shapes in images? If yes, select a technique that captures edge/gradient information
      - Am I using audio? Perhaps something that pulls our short-term frequency based features?
  - Machine Learning
    - Pass extracted features to the chosen learning technique

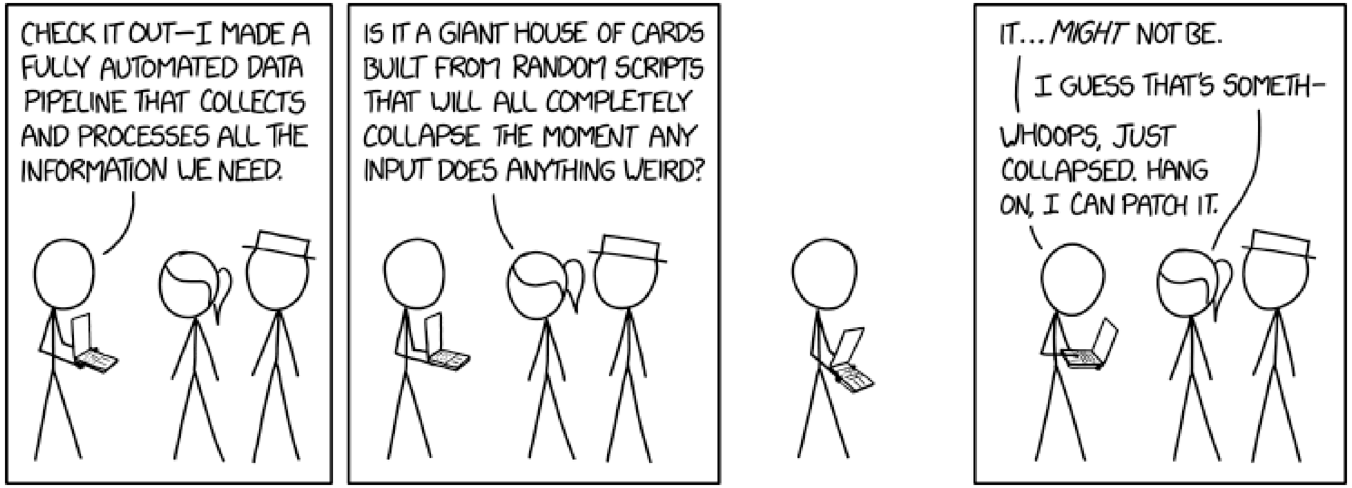
“Traditional” Machine Learning

- Feature extraction based on domain knowledge
  - Leads to “Feature engineering”: finding the optimal features for a problem/dataset
  - Can be a tedious, iterative process
  - Requires domain knowledge to extract the “best” features

- Different tasks require quite different formulations
  - Audio and Image tasks have totally different pipelines, even if the machine learning model is the same

“Traditional” Machine Learning

- Multiple steps within the pipeline for a single problem
  - May need to extract multiple sets of features
  - Machine learning component is separate
  - Lots of places for things to go wrong



Cartoon from XKCD

“Traditional” Machine Learning in CAB420

- We will look at some "traditional" methods
  - Regression, SVMs, Random Forests, etc.

- We will take a very limited look at feature extraction
  - This is a highly domain and data specific process
  - We will briefly cover some methods as they relate to specific data and examples

Neural Networks

AND HOW WE GOT TO DEEP LEARNING

Neural Networks

- A computer system modelled on the human brain and nervous system
  - Built on simple, stacked operations

- Typically have
  - High numbers of parameters
  - Lots of interconnections

- Data propagates through the networks
  - Input -> Intermediate Layers -> Output
  - Broadly similar to how we think the brain works

History

- Initial research started in the 1940's and 50's
  - Perceptron developed in 1958
  - First learnable networks soon after

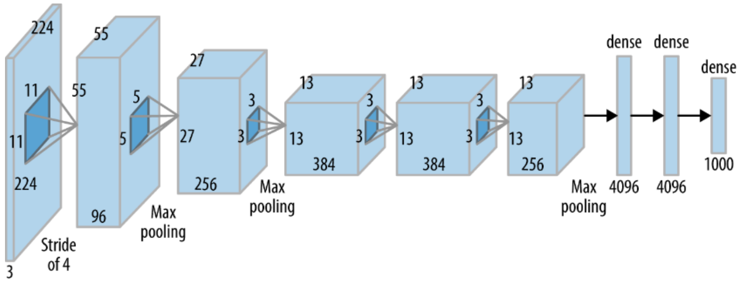
- Ongoing development through the 1980's and 90's
  - Backpropagation proposed in 1986
    - Allowed learning of complex, multi-layer, networks relatively easily
  - Recurrent Networks
    - LSTM proposed in 1997, now a mainstay of deep learning methods
  - Methods on the periphery of machine learning and optimisation

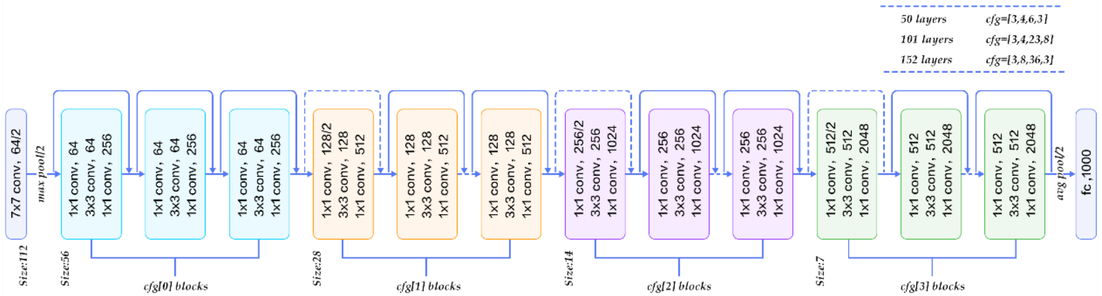
- Deep Learning
  - Emerged in the late 2000's
  - Gained widespread attention in 2012 with "Alexnet"
  - Now the dominant machine learning method across most domains

How Much Depth?

- A few years back the focus was making things as deep as possible
  - AlexNet, 8 layers
  - VGG19, 19 layers
  - ResNet, 50 to 152 layers (depending on which variant you used)

- The obsession with depth has somewhat passed now
  - Benefits diminish as we get deeper, i.e. going from 3 to 15 layers gives a huge gain; going from 100 to 500 doesn't add much.





More Deep is More Better?

- More depth gives
  - Ability to learn higher level features, so a richer representation
  - Typically more parameters, so more representative power

- But we also get
  - More parameters, which is harder to learn, and more likely to overfit
  - More layers, further for gradients to propagate
  - Harder to fit networks in memory
    - Need specialist hardware, and very long times to train
    - Not a problem for Google, a pain for the rest of us

Deep Learning Pipeline

- Common Pipeline:
  - Pre-processing
    - Resizing, filtering, noise reduction
      - Input images (usually) need to be the same size
  - Deep Learning
    - Pass raw data to network
    - Let the network learn it's own representation and make the decision in one pass

Deep Learning: Pros

- Easy to extend model to multiple tasks, or adapt to new tasks
  - Change the loss function; or
  - Add an extra loss
  - Fine tune models

- No feature engineering
  - Let the model learn the features
  - Can lead to more robust representations

- State of the art performance for most (all?) tasks

Deep Learning: Cons

- Models are huge compared to “traditional” approaches
  - Millions vs Hundreds of parameters
  - Requires more data, runtime, memory
  - Models are harder to interpret
    - Explainable AI. Why did my model make this decision?

- No feature engineering, instead we have network engineering
  - What layers, how many, what parameters, etc.
  - Can be very slow to iterate over these decisions, or simply may not be possible to determine a truly “optimal” solution

CAB420: Linear Regression

OUR FIRST ML APPROACH

Simple Linear Regression

- Often, we want to find a relationship between two variables to:
  - Predict behaviour
  - Explore relationships

- The simplest way to do this is to assume a linear relationship

$$y = \beta_0 + \beta_1 x$$

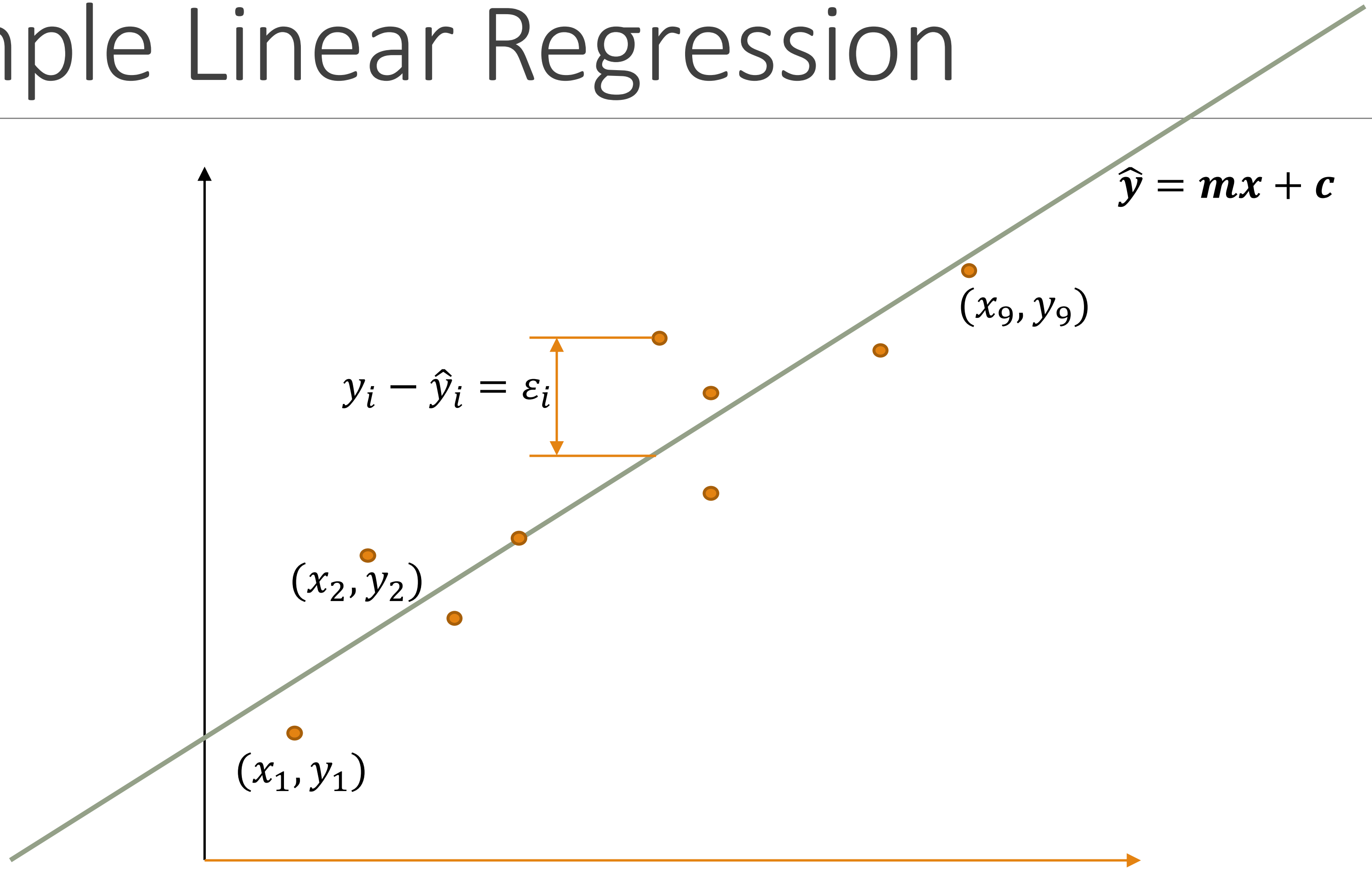
- Often, this assumption is reasonable and a good starting point.

Simple Linear Regression

- For points (x, y) on a line $y = mx + c$ we have the relationship
  - $y_1 = mx_1 + c$
  - $y_2 = mx_2 + c$
  - …
  - $y_n = mx_n + c$

- In practice, this is rarely (if ever) a perfect fit to our data
- But what if our data aren't all on a straight line together?
- We propose that our points should be on a line.
  - $\hat{y} = mx + c$

Simple Linear Regression



Simple Linear Regression

- Point *i* in our input data is given by
$$y_i = mx_i + c + \varepsilon_i$$
- The (vertical) distance to where a point *should* be, $\hat{y}_i$, according to the straight-line model, and where it *is*, y_i, is called the **error**, ε_i.
- We assume $\varepsilon_i \sim N\left(0, \sigma^2\right)$
- We want to minimise the distance from all the observed y_i to all the $\hat{y}_i$
- Find the values of *c* and *m* that minimise the sum of the square of the errors,
$$\sum_{i=1}^{n} \varepsilon_i^2 = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{n} (y_i - (mx_i + c))^2$$

Simple Linear Regression

- **Simple** Linear Regression
  - We have only one **x**

$$y = \beta_0 + \beta_1 x_1$$

- Need to estimate β_0 and β_1 given a set of observations, $\boldsymbol{y} = (y_1, \dots, y_n)$ and $\boldsymbol{x} = (x_1, \dots, x_n)$
  - Need at least as many observations as we have terms to estimate
  - Ideally, our number of observations is much larger than the number of terms we are trying to predict.

Simple Linear Regression

- We expect each prediction to have a small error (i.e. our model won’t be perfect)

$$y_i - \beta_0 - \beta_1 x_1 = \varepsilon_i$$

  - ε_i is the residual

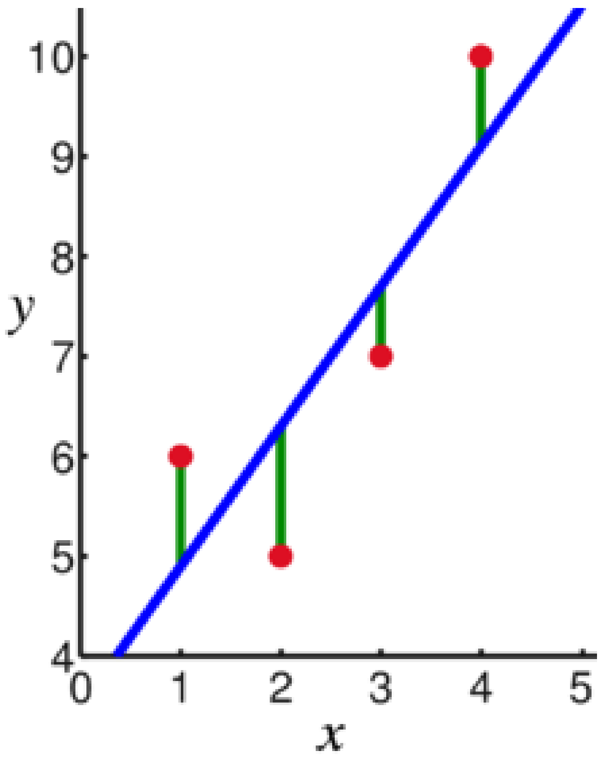
- We assume:
  - Residuals are independent, identically distributed random variables
  - Residuals follow a Gaussian distributed around 0
- From this, we can solve directly for β_0 and β_1

How is this done?

- Least Squares Fitting:
  - We try to find the line that minimises

$$S = \sum_{i=1}^{n} (y_i - y_i')^2$$

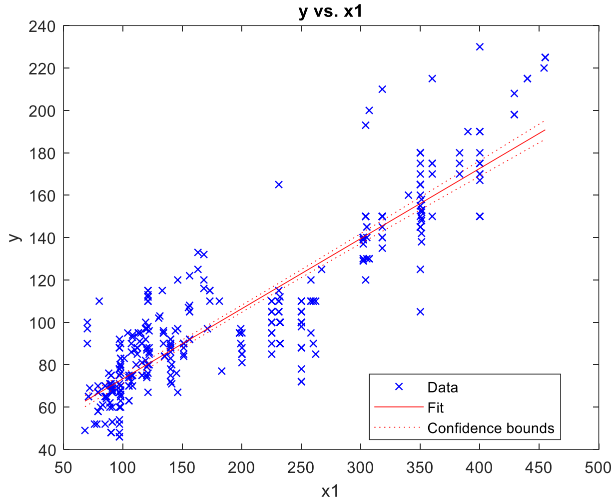
  - y_i are the true values
  - y_i' are the predicted values

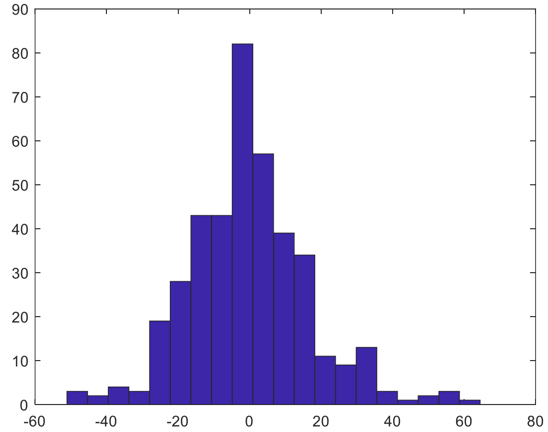


Least Squares Fitting

- Given this we expect
  - Our fitted line to go “though the middle” of the points
  - Our errors (the residuals) to be symmetric

- Note that this has a closed form solution
  - i.e. can be solved directly





Correlation

AND HOW IT CAN HELP US WITH REGRESSION

A Definition: Correlation

noun: a mutual relationship or connection between two things

Or

- How much two things change together

Measures a linear relationship between two data series

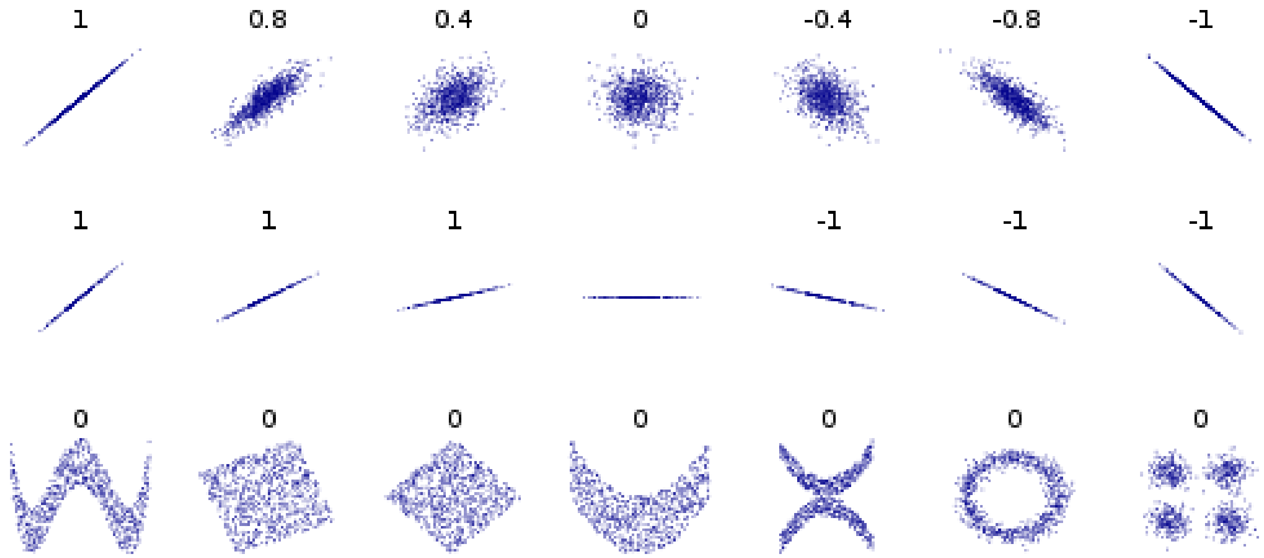
- If things are correlated, we can use regression to predict one from the other

Correlation - Formulation

- We’re considering Pearson’s Correlation Coefficient
- We first need to know our co-variance
  - $\sigma_{xy} = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu_x)(y_i - \mu_y)$
- From which we can derive the correlation
  - $\rho_{xy} = \frac{\sigma_{xy}}{\sigma_x \sigma_y}$
- We can also get the correlation for a sample (rather than the population)
  - $s_{xy} = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})$
  - $r_{xy} = \frac{s_{xy}}{s_x s_y}$

Correlation Coefficient

- Ranges from -1 to +1
  - -1, as one variable increases the other decreases
  - 0, statistically independent
  - +1, increase and decrease together



Anscombe's quartet

- Four sets of (x, y) data
  - Means of x = 9
  - Means of y = 7.5
  - Standard deviations of x = 3.3
  - Standard deviations of y = 2.0
  - Correlations = 0.81

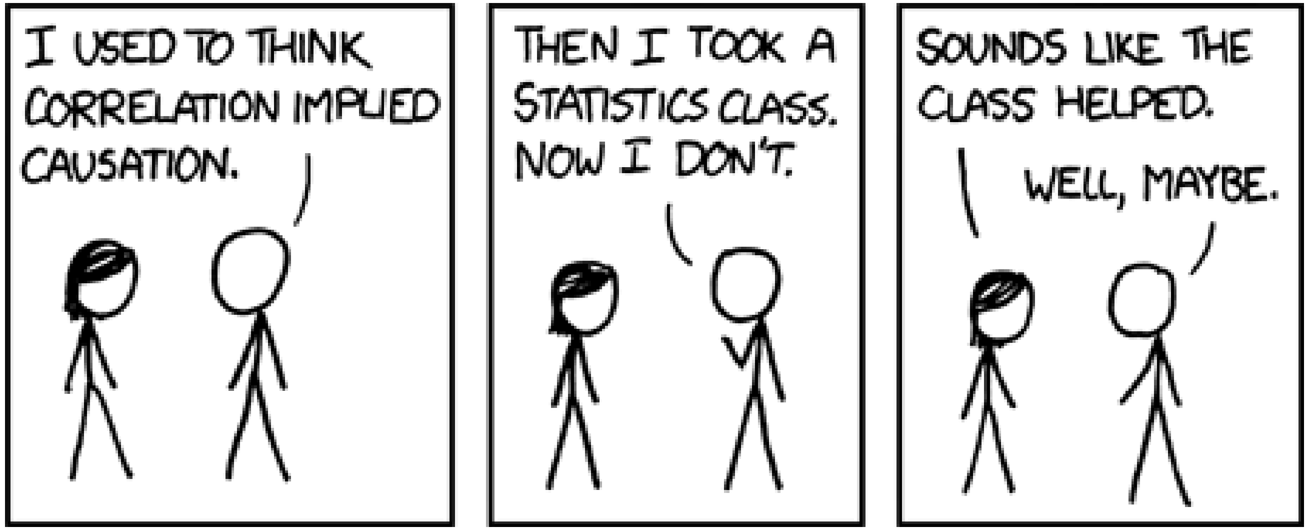
Anscombe's quartet

y

x

Correlation vs Causation

From XKCD



Correlation: Why do we care?

- Ideally, we want
  - Our predictors to be correlated with the response
    - i.e. there is a linear relationship there to be modelled
  - Our predictors to be uncorrelated with each other
    - i.e. each predictor models a different aspect of the overall relationship

- If we have correlated predictors, we can end up with redundancy in the model and unexpected p-values

Multiple Linear Regression

PREDICTING ONE THING FROM SEVERAL THINGS

Multiple Linear Regression

- Still fitting a line to some data
  - Just in multiple dimensions
- Our line is now:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p$$

or

$$y = \beta_0 + \sum_{i=1}^{p} \beta_i x_i$$

  - We have many predictors (x_1, x_2, x_3, …)
  - We have one response (y)
  - We need to find $\beta_0, \beta_1, \beta_2, \dots, \beta_p$
  - We also want our number of samples, n, to be much larger than the number of coefficients
    - This becomes harder as we add more terms

Multiple Linear Regression: Matrix Formulation

- Linear regression can be seen as a single equation involving matrices and vectors. For example, with two explanatory variables and *n* points of data:

$$\mathbf{y} = \mathbf{x}\beta + \varepsilon$$

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \mathbf{x} = \begin{bmatrix} 1 & x_{1,1} & x_{1,2,} \\ 1 & x_{2,1} & x_{2,2} \\ \vdots & \vdots & \vdots \\ 1 & x_{n,1} & x_{n,2} \end{bmatrix}, \beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{bmatrix}, \varepsilon = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}$$

- Error vector here is $\varepsilon = \mathbf{y} - \mathbf{x}\beta$, which is a function of β.

Multiple Linear Regression: Matrix Formulation

- We want to minimise the **sum of squared errors**:

$$SSE(\beta) = \sum_{i=1}^{n} \varepsilon_i^2 = \varepsilon'\varepsilon = (\mathbf{y}'\mathbf{y} - 2\beta'\mathbf{x}'\mathbf{y} + \beta'\mathbf{x}'\mathbf{x}\beta)$$

- The way a function changes with respect to certain variables can be calculated using the function's **derivative**
- Once the derivative is calculated, set the equation to be equal to zero and solve for the same variable in order to find maximum(s) and/or minimum(s) of the function.

Multiple Linear Regression: Matrix Formulation

- Sum of squared errors term:
$$SSE(\beta) = (\mathbf{y}'\mathbf{y} - 2\beta'\mathbf{x}'\mathbf{y} + \beta'\mathbf{x}'\mathbf{x}\beta)$$

- Derivative of SSE with respect to β:
$$\nabla SSE(\beta) = 2(\mathbf{x}'\mathbf{x}\beta - \mathbf{x}'\mathbf{y})$$

- Setting to 0 and solving for β gives the optimal vector, $\hat{\beta}$:
$$\hat{\beta} = (\mathbf{x}'\mathbf{x})^{-1}\mathbf{x}'\mathbf{y}$$

- Linear regression is also known as **ordinary least squares**

Multiple Linear Regression: Demo

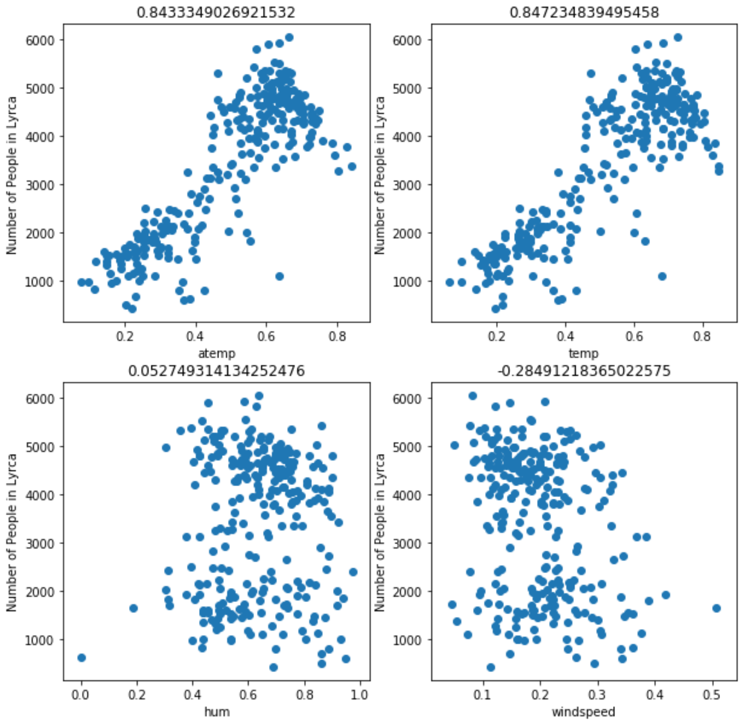
- See ***CAB420_Regression_Example_1_Linear_Regression.ipynb***
- Task:
  - Fit a linear regression model to predict cyclist numbers from weather data
    - Temperature, apparent temperature, humidity, windspeed
  - Evaluate the model, and explore the impact of each variable and how they contribute the to the model
- This demo will be covered in more detail in the interactive lecture session

Regression Demo: High Level Process

- Load and visualise data
  - Looking for missing data, errors, exploring relationships (i.e. correlation) in the data
- Split data
  - Train and test sets
- Extract predictors (inputs) and response (output)
- Fit model
- Explore model performance
  - Quality of fit, validity of assumptions
- Refine model
  - And explore performance, refine again, etc.

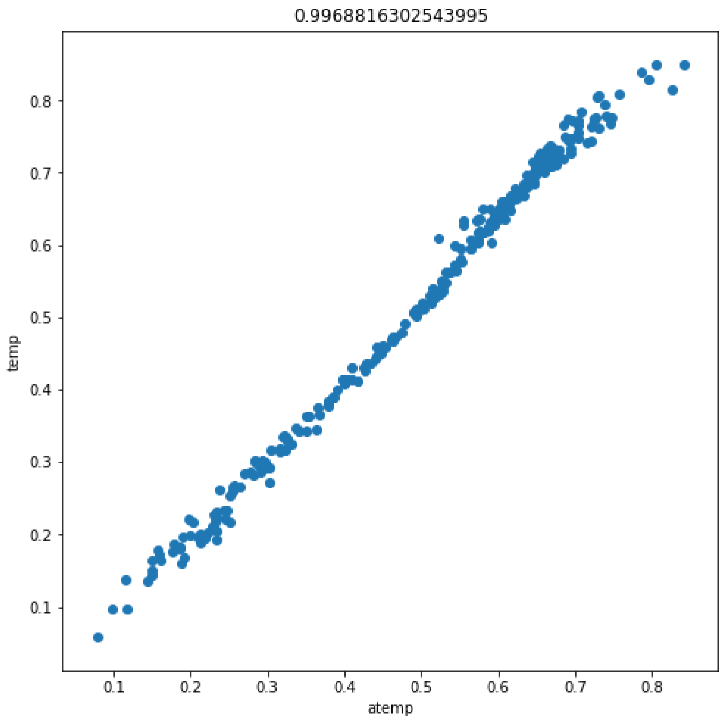
Correlation between predictors and response

- temp and atemp highly correlated with our response (cnt)
- hum and windspeed are less correlated with the response (cnt)



Correlation between predictors

- temp and atemp are very highly correlated



A Simple Model

Fitting this:

- `cnt ~ atemp + temp + hum + windspeed`

We get this output:

```
OLS Regression Results
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.748
Model:                            OLS   Adj. R-squared:                  0.744
Method:                 Least Squares   F-statistic:                     198.4
Date:                Mon, 11 Jan 2021   Prob (F-statistic):           8.06e-79
Time:                        11:36:23   Log-Likelihood:                -2190.2
No. Observations:                 273   AIC:                             4390.
Df Residuals:                     268   BIC:                             4408.
Df Model:                           4
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept   1708.8451    296.956      5.755      0.000    1124.182    2293.509
atemp      -3132.5570   3164.040     -0.990      0.323   -9362.093    3096.979
temp        8823.6644   2823.617      3.125      0.002    3264.372    1.44e+04
hum        -1134.9410    302.778     -3.748      0.000   -1731.067    -538.815
windspeed  -3052.9184    642.368     -4.753      0.000   -4317.647   -1788.190
==============================================================================
Omnibus:                       15.311   Durbin-Watson:                   0.963
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               22.768
Skew:                          -0.383   Prob(JB):                     1.14e-05
Kurtosis:                       4.190   Cond. No.                         132.
==============================================================================
```

Is any good?

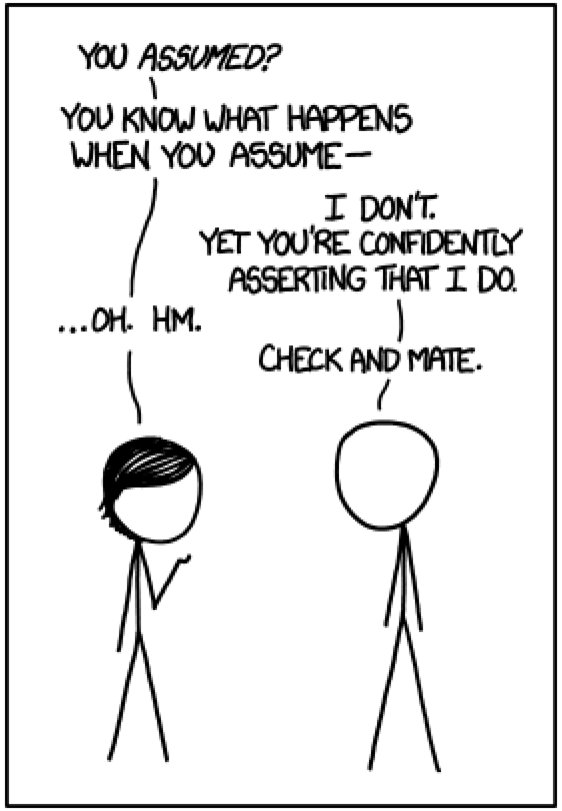
- How do we tell if our model any good?
  - Does it fit the data?
    - Look at R^2, RMSE on the training data
  - Can it make accurate predictions?
    - Look at RMSE on the test set (or unseen data)
  - Are our terms significant?
    - Check the p-values
  - Does it violate any of our assumptions?

Things Linear Regression Assumes

SOMETHING ABOUT A DONKEY?

Assumptions

- Linearity
  - The relationship between our predictors and response is linear
- Independence
  - Observations are independent from each other
- No Multicollinearity
  - Predictors are not correlated with each other
- Normality
  - Errors follow a normal (Gaussian) distribution
- Homoscedasticity
  - The variances of the errors (the residuals) is consistent across the values of the variables
- No endogeneity
  - No relationship between errors and predictor variables



From XKCD

Assumptions – Do we care?

- If we violate these assumptions our model may be unreliable, or perform poorly
- You may see
  - Poor estimations in some or all portions of the data space
  - Poor performance on unseen data
  - Model terms that don’t contribute helpfully to the model
  - High sensitivity to noise
- Thankfully, we can test our assumptions and use this to improve our model

Regression Diagnostics

CAN WE FIX IT? YES, WE CAN!

Regression Diagnostics (Testing Our Assumptions)

- We'll use four main tools to test our assumptions:
  - Plots of predictions vs actual values
    - Look for agreement between what our model predicts, and what should happen
  - Regression output
    - P-values, R^2 and RMSEs
  - Correlation
    - Between pairs of predictors (look for multi-collinearity) and predictors and response (look for linearity)
  - Analysis of residuals (model errors)
    - Plots to test for normality of residuals
    - Plots to test for Homoscedasticity

Analysing Model Output

- Typically looks something like this:

```
OLS Regression Results
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.748
Model:                            OLS   Adj. R-squared:                  0.744
Method:                 Least Squares   F-statistic:                     198.4
Date:                Mon, 11 Jan 2021   Prob (F-statistic):           8.06e-79
Time:                        11:36:23   Log-Likelihood:                -2190.2
No. Observations:                 273   AIC:                             4390.
Df Residuals:                     268   BIC:                             4408.
Df Model:                           4
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept   1708.8451    296.956      5.755      0.000    1124.182    2293.509
atemp      -3132.5570   3164.040     -0.990      0.323   -9362.093    3096.979
temp        8823.6644   2823.617      3.125      0.002    3264.372    1.44e+04
hum        -1134.9410    302.778     -3.748      0.000   -1731.067    -538.815
windspeed  -3052.9184    642.368     -4.753      0.000   -4317.647   -1788.190
==============================================================================
Omnibus:                       15.311   Durbin-Watson:                   0.963
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               22.768
Skew:                          -0.383   Prob(JB):                     1.14e-05
Kurtosis:                       4.190   Cond. No.                         132.
==============================================================================
```

A quick aside: p-values

- The result of a statistical test and represent the probability, under the assumption of no effect or no difference (null hypothesis) of obtaining a result equal to or more extreme than what was observed
- Essentially, they measures how likely something is due to chance alone
- A small p-value means what is observed is very likely due to something other than chance
- For our regression models, we will get p-values for
  - The model as a whole
  - Individual terms

Whole of Model Analysis

- R-Squared: Proportion of observed variance explained by the model. A value of 1 means the model explains everything, a value of 0 means it explains nothing. Defined as follows:
  - $R^2 = 1 - \frac{SSE}{SSY}$
  - $SSE = \sum_{i=0}^{n}(y_i - \hat{y}_i)$
  - $SSY = \sum_{i=0}^{n}(y_i - \bar{y})$; $\bar{y}$ = mean of the data
- Adjusted R-Squared: Considers the model complexity (number of terms) alongside how much variance it explains.
  - $R^2_{Adj} = 1 - \frac{n-1}{n-p-1}\frac{SSE}{SSY}$
- F-statistic: Test statistic for the whole model. A p-value is also provided to indicate if the entire model is significant.

```
OLS Regression Results
==============================================================================
Dep. Variable:                    cnt   R-squared:                       0.748
Model:                            OLS   Adj. R-squared:                  0.744
Method:                 Least Squares   F-statistic:                     198.4
Date:                Mon, 11 Jan 2021   Prob (F-statistic):           8.06e-79
Time:                        11:36:23   Log-Likelihood:                -2190.2
No. Observations:                 273   AIC:                             4390.
Df Residuals:                     268   BIC:                             4408.
Df Model:                           4
Covariance Type:            nonrobust
==============================================================================
```

Whole of Model Analysis

- Number of observations: how many samples we have, in general, more is better (to a point, things get trickier when we can't fit the data in memory)
- Error degrees of freedom: Number of observations minus the number of terms. We want this to be big.

- Root Mean Squared Error: A measure of error for the model. Smaller is better (not in Python's StatsModels output, though easy to calculate).

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=0}^{n}(y_i - \hat{y}_i)^2}$$

OLS Regression Results

Dep. Variable:	cnt	R-squared:	0.748
Model:	OLS	Adj. R-squared:	0.744
Method:	Least Squares	F-statistic:	198.4
Date:	Mon, 11 Jan 2021	Prob (F-statistic):	8.06e-79
Time:	11:36:23	Log-Likelihood:	-2190.2
No. Observations:	273	AIC:	4390.
Df Residuals:	268	BIC:	4408.
Df Model:	4		
Covariance Type:	nonrobust		

Whole of Model Analysis

- RMSE is only really useful to compare models trained on the same data
  - The range of RMSE depends on the data
- R-squared and adjusted R-squared are good indicators of predictive power
  - Can only calculate this data on the training set
  - Based around the assumptions made for residuals and their distribution
  - R-squared is ALWAYS calculated on training data
    - Blindly maximising this can lead to overfitting

Analysing Model Terms

- We have
  - coef: the actual coefficient value, i.e. $\beta_0, \beta_1, \beta_2, \ldots, \beta_p$
  - std err: standard error, a measure of how much the coefficient changes by if we resample the data and recompute the regression
  - t: $\frac{Estimate}{SE}$, we want this to be a long way from 0, as it indicates that the term is less likely to be the result of noise
  - P>|t|: p-value for the null hypothesis that the coefficient is equal to 0. Low p-value means that you can reject the null hypothesis, i.e. that the term is significant.

	coef	std err	t	P>|t|
Intercept	1708.8451	296.956	5.755	0.000
atemp	-3132.5570	3164.040	-0.990	0.323
temp	8823.6644	2823.617	3.125	0.002
hum	-1134.9410	302.778	-3.748	0.000
windspeed	-3052.9184	642.368	-4.753	0.000

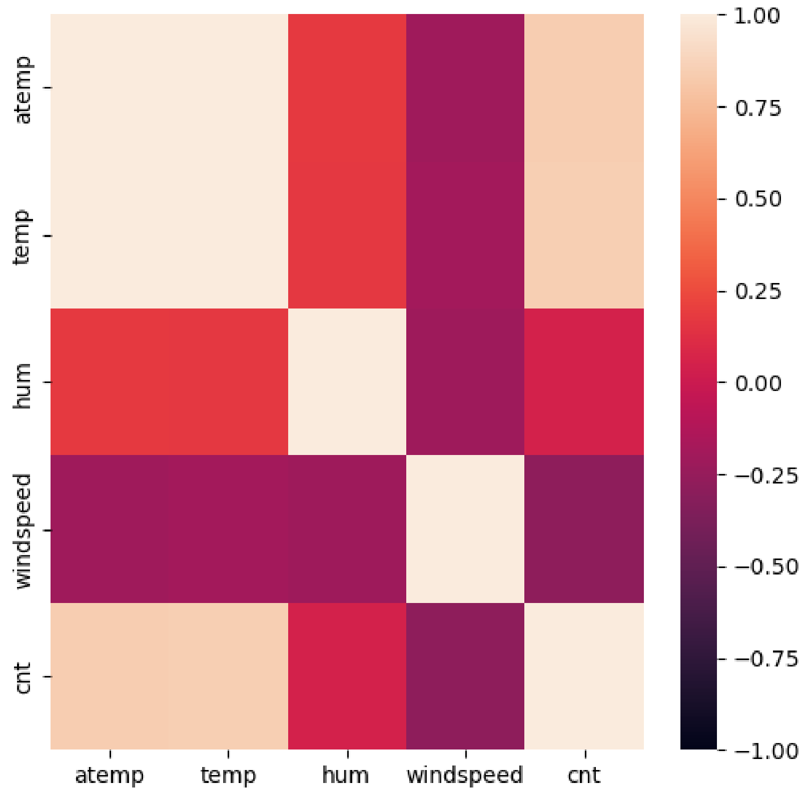
Analysis Model Terms

- Focus on the p-values
  - High p-value indicates a term that is not significant
    - Does not contribute to the model
  - Why does it not contribute?
    - There could be no actual relationship
      - Check correlation with response
    - Could have correlated variables
      - Two or more predictor variables capture the same relationship with the response
      - Check correlation with other predictors
  - Remove poor terms one by one until all terms are significant

	coef	std err	t	P>|t|
Intercept	1708.8451	296.956	5.755	0.000
atemp	-3132.5570	3164.040	-0.990	0.323
temp	8823.6644	2823.617	3.125	0.002
hum	-1134.9410	302.778	-3.748	0.000
windspeed	-3052.9184	642.368	-4.753	0.000

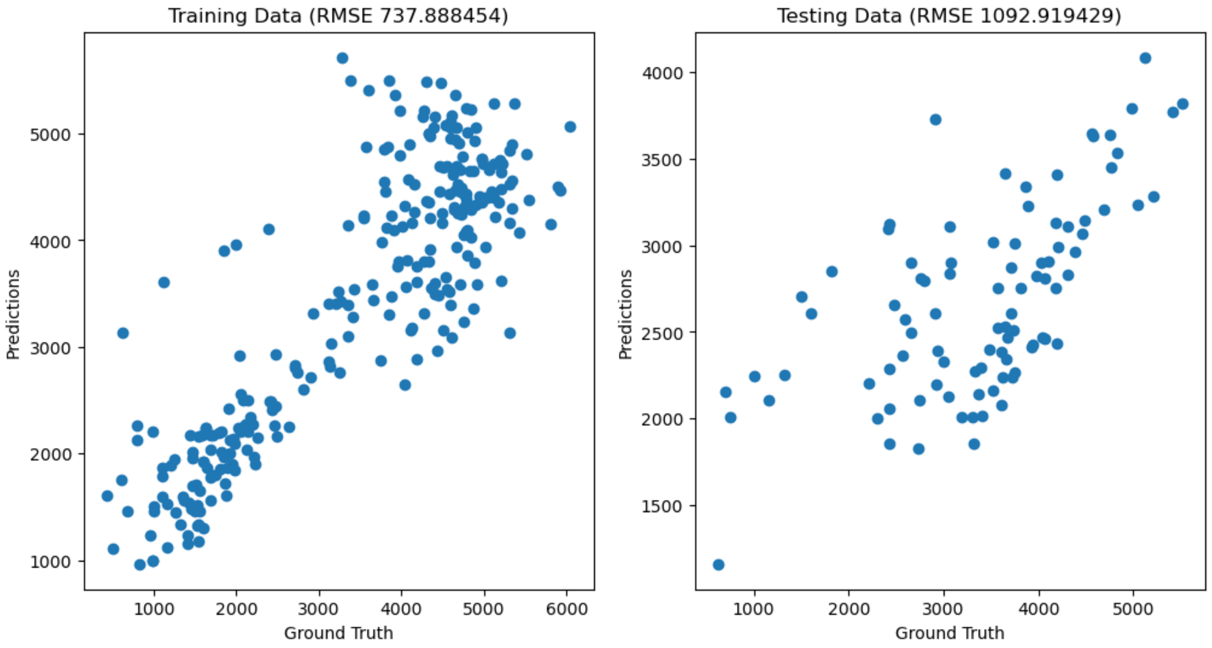
Correlation

- The heatmap shows correlation between
  - Pairs of predictors
  - Predictors and the response
- This allows us to quickly look for problems across a lot of variables
- We can see that temp and atemp are extremely highly correlated
- All terms have at least some correlation with the response
  - Though humidity has a fairly weak relationship



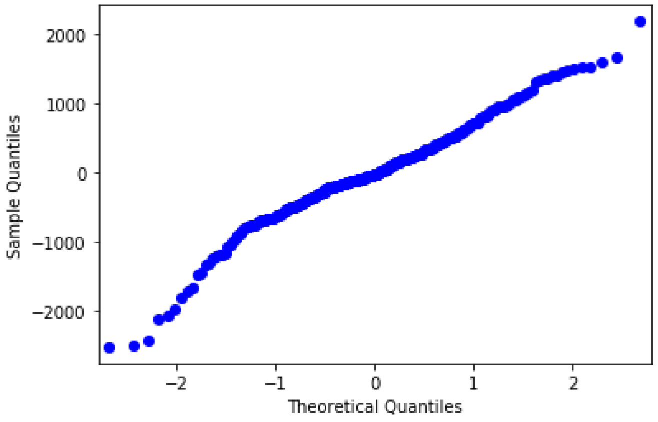
Analysing Predictions

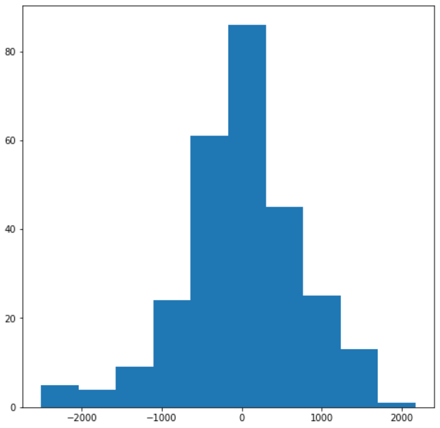
- Scatter plot of ground truth vs predictions
  - Ideally, we see a nice straight line, or at least something mostly linear
  - Weird shapes (like curves) would suggest something non-linear
  - These look ok

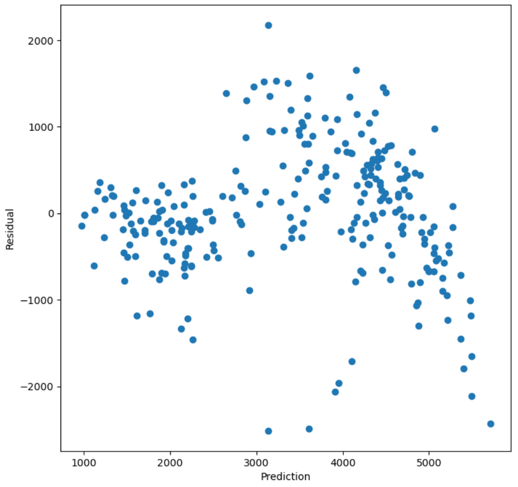


Analysing Residuals

- Our residuals should
  - Follow a Gaussian Distribution
    - We can use a qqplot (left) and see how many points are away from the normal distribution line
    - We can use a histogram of residuals (middle) and look for a normal distribution
  - Have a constant variance across the range of data
    - Scatter plot of predictions vs residuals (right)
  - Our residuals look close-to Gaussian, but the variance perhaps is not quite constant







Improving the Model

- atemp and temp are very similar variables
  - High correlation
  - Impacts fitted terms and p-values
- Remove one of the terms
  - See example script for detailed results

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	1708.8451	296.956	5.755	0.000	1124.182	2293.509
atemp	-3132.5570	3164.040	-0.990	0.323	-9362.093	3096.979
temp	8823.6644	2823.617	3.125	0.002	3264.372	1.44e+04
hum	-1134.9410	302.778	-3.748	0.000	-1731.067	-538.815
windspeed	-3052.9184	642.368	-4.753	0.000	-4317.647	-1788.190

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	1598.9598	275.424	5.805	0.000	1056.698	2141.222
temp	6037.2059	227.171	26.576	0.000	5589.946	6484.466
hum	-1144.5128	302.612	-3.782	0.000	-1740.303	-548.723
windspeed	-2966.6479	636.407	-4.662	0.000	-4219.619	-1713.677

Other Considerations

- Data splits
  - Random vs Sequential
- Categorical Variables
  - Specified and handled differently by the model
  - Each category has an "offset" associated with it that is included when that category is "on"
  - Can rapidly increase model size
    - Category with 20 options will add 19 new terms
  - Can cause problems when some categories are rare
- See example script for details of both these

Other Considerations

- Higher order terms
  - Product of two predictors, square of a predictor, higher order polynomials relating predictors
  - Can greatly improve model performance
    - i.e. can now capture quadratic relationships
  - Can cause overfitting
    - Lots of additional terms can be added very quickly
    - Care needs to be taken to ensure all terms are meaningful and data is sufficient

- See example script for details