CAB420: Ethics and ML

WITH GREAT POWER, COMES GREAT RESPONSIBILITY

Machine Learning in Society

- We are seeing rapid adoption of ML in society for several applications
- Some of these applications are fairly innocuous
    - Predictive text when sending messages
    - Advertising recommendations
- Some, are a bit more serious
    - Border control
    - Job application pre-screening
    - Surveillance and monitoring technologies
- For many we don’t quite know what the impacts are
    - Chat-GPT, DALL-E
- Consider the implications of these systems
    - All can be used “for good”
    - All can have negative impacts, though some more severe than others

Machine Learning in Society

- What makes a machine learning system suitable for use?
  - What level of accuracy is needed?
    - Does the level of accuracy change depend on the application?
    - If so, how?
  - What happens when a machine learning system “gets it wrong”?
  - Who’s at fault when a system makes an error?
  - Is the system biased?
  - Should the impacts of a system be considered before it’s released upon the world?

- At present, there is
  - No single set of standards or guidelines to help with the above questions
  - Little to no external oversight regarding any deployed ML model
  - Little, if any, consideration to the impacts of systems on others

Machine Learning and Errors

- Errors can be caused by many factors
  - Issues with training data
    - Not enough diversity, incorrect labels, too little data, too little curation
  - Issues with model design
    - Too simplistic, too complex, invalid assumptions
  - There may also be genuine outliers that lead to an error
    - Or is this just another issue with a lack of diversity?
    - Can an ML system ever cover all possibilities?

Dealing with Errors in Practice

- Ideally, we'd review decisions made by an ML system (human in the loop), but
  - Is this practical?
  - Will the human be more accurate?
- Could only "unsure" samples be reviewed?
  - Reporting model confidence is not straight forward
    - Particularly for deep learning
  - Model decisions can still be confident and be totally wrong
- Mechanisms to deal with incorrect decisions are limited, if present at all
- Within chat/text systems, "blacklisting" is often used
  - Simple rules to avoid certain topics, but…
    - Siri blacklisted topics for healthcare, yet "she could tell you where to hide a body"
    - Chat-GPTs blacklisted topics can be bypassed quite easily
      - https://www.lesswrong.com/posts/7fYxxtZqjuYXhBA2D/testing-ways-to-bypass-chatgpt-s-safety-features

ML and Bias

- Bias is a large concern in ML models that are deployed
- Data Bias
  - Caused by using limited or flawed data in a model
    - Very hard to avoid totally, all data sets are a sample
    - Amazon’s hiring tool that was biased against women (https://www.reuters.com/article/us-amazon-com-jobs-automation-insight/amazon-scraps-secret-ai-recruiting-tool-that-showed-bias-against-women-idUSKCN1MK08G)
    - Hate speech detection biased against black people (https://futurism.com/the-byte/google-hate-speech-ai-biased)
- Technical and contextual biases
  - Design choices/flaws that disadvantage one group over another
  - Omitting information that provides context for the data and decision

ML and Data

- ML depends on data, but data needs to be sourced and used ethically
- Ever, a cloud storage photo app, illegally used user photos to train face recognition models, and was ordered to delete the resultant models
    - https://techcrunch.com/2021/01/12/ftc-settlement-with-ever-orders-data-and-ais-deleted-after-facial-recognition-pivot/
- Duke MTMC dataset, a large multi-target multi-camera object tracking and person re-id dataset, was heavily used by the Chinese military and Chinese surveillance companies, and was subsequently removed.
    - https://exposing.ai/duke_mtmc/
- Training on data collected and annotated in an automated way is common
    - GPT-3 is trained by data pulled from the internet
        - Including copywritten material, and posts on Reddit
        - DALL-E is trained in a similar way, but with images
    - 80 Million Tiny Images dataset uses labels automatically generated from WordNet
        - Contains harmful, racist and derogatory terms as labels
        - https://openreview.net/pdf?id=s-e2zaAlG3I
- Within academia, it is common to mine social media for data
    - Youtube, flickr, twitter, etc.
    - Commonly seen as “fair use”, though sharing such data repositories is murkier

ML and Ethics

- Increasing awareness of ethical issues relating to ML
  - More coverage in academic literature, including dedicated events (https://dl.acm.org/doi/proceedings/10.1145/3531146)
- Companies are beginning to enact their own policies
  - Microsoft (https://www.microsoft.com/en-us/ai/responsible-ai?activetab=pivot1%3aprimaryr6)
- But often only when convenient
  - Google firing their Ethics team leader after they pointed out various issues with machine learning systems
  - https://www.theverge.com/22309962/timnit-gebru-google-harassment-campaign-jeff-dean
- Several guidelines/principals for ethical use of AI have been developed by governments and public bodies in the past couple of years
  - Australian Government (https://www.industry.gov.au/data-and-publications/building-australias-artificial-intelligence-capability/ai-ethics-framework/ai-ethics-principles)
  - Institute for Ethical AI and Machine Learning (https://ethical.institute/principles.html)

AI Ethics Principals (Australian Government)

- Human, social and environmental wellbeing
  - Throughout their lifecycle, AI systems should benefit individuals, society and the environment.
- Human-centred values
  - Throughout their lifecycle, AI systems should respect human rights, diversity, and the autonomy of individuals.
- Fairness
  - Throughout their lifecycle, AI systems should be inclusive and accessible, and should not involve or result in unfair discrimination against individuals, communities or groups.
- Privacy protection and security
  - Throughout their lifecycle, AI systems should respect and uphold privacy rights and data protection, and ensure the security of data.

AI Ethics Principals (Australian Government)

- Reliability and safety
  - Throughout their lifecycle, AI systems should reliably operate in accordance with their intended purpose.

- Transparency and explainability
  - There should be transparency and responsible disclosure to ensure people know when they are being significantly impacted by an AI system, and can find out when an AI system is engaging with them.

- Contestability
  - When an AI system significantly impacts a person, community, group or environment, there should be a timely process to allow people to challenge the use or output of the AI system.

- Accountability
  - Those responsible for the different phases of the AI system lifecycle should be identifiable and accountable for the outcomes of the AI systems, and human oversight of AI systems should be enabled.

Related AI/ML Ethics Issues

- There are other questions around ethics and AI/ML
- Rapidly emerging issues
  - Intellectual property and copywrite challenges
    - Training models on copywrite data, and to mimic the artistic style of others
  - Job loss and wealth inequality
    - What happens when people's jobs are automated?
  - Military applications of AI/ML
  - Ongoing rollout of surveillance systems
- Potential Future issues
  - The singularity and maintaining control over AIs
  - How should we treat AIs? What rights would they have?

Ethics and ML/AI

- Ethical and equitable development and deployment of ML systems is challenging and complex issue, and is **not just a tech problem**
- Greater consideration is needed about how we deploy and use models, and their broader implications
- Eliminating data bias totally may never be possible
  - But we can be clear about limitations
- We can strive to include uncertainty estimates in models
  - Though this is on-going research, and we're not there yet
- We can provide a "human in the loop" to help review and improve models
- Ethics in AI/ML is an evolving discussion
  - If you are keen to continue in ML, I'd encourage you to read further about these issues

CAB420: Regression Summary

WHAT WAS REGRESSION AGAIN?

Regression

- Predicting a continuous output from one or more inputs
  - There are methods to deal with non-continuous outputs, but these are outside of CAB420’s scope

- We’ve considered
  - Linear Regression
  - Regularised Linear Regression
    - Ridge (L2) Regularisation
    - LASSO (L1) Regularisation
  - Neural Networks

Linear Regression

- Learning a “line of best fit”
- Models a linear relationship between the inputs (predictors) and output (response)

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_j x_j$$

- To find the “line of best fit”, we seek to minimise the sum of squared differences between the actual points and the predicted points (least squares regression)

$$\sum_{i=1}^{n} \left(y_i - \sum_{j} x_{ij} \beta_j \right)^2$$

- Several assumptions
  - Samples are independent
    - This may not be the case if we have time-series data
  - Predictors are independent
    - Often, we’ll have some correlation between predictors and violate this assumption
  - Residuals follow a Gaussian distribution
    - Can test by looking at a histogram of residuals, or a qq-plot
- Ideally, we have more samples than predictors

Regularised Regression

- Adds a penalty (regularisation) term to our objective function

$$\sum_{i=1}^{n}\left(y_i - \sum_{j} x_{ij}\beta_j\right)^2 + \lambda P$$

- The form of the penalty term determines the type of regularisation
- L2 penalty is Ridge regression
  - Penalty term is the sum of the model coefficients squared

$$\sum_{i=1}^{n}\left(y_i - \sum_{j} x_{ij}\beta_j\right)^2 + \lambda \sum_{j=1}^{p}\left\|\beta_j\right\|_2$$

- L1 penalty is LASSO regression
  - Penalty term is the sum of the absolute values of the model coefficients

$$\sum_{i=1}^{n}\left(y_i - \sum_{j} x_{ij}\beta_j\right)^2 + \lambda \sum_{j=1}^{p}\left\|\beta_j\right\|_1$$

Regularised Regression

- Regularisation imposes additional constraints on the model
  - Constraints are not based on accuracy, but complexity
  - Constraints have a weight, λ, that we need to set
- Constraints encourage smaller model coefficients
  - Smaller coefficients mean that that the model is less sensitive to small changes in the input
  - Can help curb overfitting, but will reduce accuracy on the training set
- Use the validation set to find the optimal value of λ
  - Train the model on the training set
  - Evaluate the model on the validation set
  - Find the value of λ that leads to best accuracy on the validation set
- Ridge and LASSO will both lead to smaller coefficients as λ increases, but
  - Ridge will bring coefficients increasingly close to 0, but they will never reach 0
  - LASSO will eliminate terms (set coefficients to 0), and produce a constant model (all terms 0) once a sufficiently large λ is reached
- Standardisation often used with regularised regression
  - Helps avoid issues caused by scale variation in the data
  - Eliminates the constant term, β_0, from the model

Regression with Deep Neural Networks

- Deep networks can be used for regression
- For regression, the network should
    - Use an output activation appropriate for the task
        - If the regression output is unbounded (i.e. can range from $[-\infty \ldots +\infty]$), then no activation is appropriate
        - If it output is positive (i.e. $[0 \ldots \infty]$), then a ReLu makes sense
    - Use an appropriate loss
        - Mean Squared Error is the sum of squared difference, the same thing that is minimised in regular least-squares regression
        - Mean Absolute Error may also be appropriate depending on the problem

An Example

- See ***CAB420_Summary_1_Regression.ipynb***

- Two regression tasks
  - Predict the price of diamonds given some properties of the diamonds
  - Predict the year a song was released given some properties of the songs

- For each task, apply
  - Linear regression
  - Ridge and LASSO regularised regression
  - A Deep Neural Network
    - Not a DCNN, so no convolutions
    - Data is tabular data, no spatial relationships for convolution to exploit

- Review this example in your own time
  - Ask questions in class or via email/slack

CAB420: Classification Summary

SOMETHING ABOUT CLASSES?

Classification

- Assigning an input to one of a predefined set of categories (or classes)
  - All classes need to be known in advance, and be present during training
  - Classifiers will tend to favour classes for which there is more data

- We've considered
  - Support Vector Machines (SVMs)
  - K Nearest Neighbours Classifiers (CKNN)
  - Random Forests
  - Deep Neural Networks

Non-Deep Learning Methods

OLD SCHOOL MACHINE LEARNING

Support Vector Machines

- Binary Classifier
- Learns a decision boundary between two **linearly separable** classes
  - But, if you need machine learning, it’s probably not linearly separable
- So, we can
  - Relax our constraints via a slack variable
  - Map to a higher dimensional space where things might be linearly separable via kernels
    - Care must be taken with kernel selection and parameters
  - Or do both
- In general
    - Good with high-dimensional and/or sparse data
    - Can be difficult to train with large (10,000 +) datasets
    - Binary in nature (though model ensembles can enable use on multi-class problems)

Support Vector Machines Parameters

- C (sometimes referred to as the box-constraint), which essentially provides regularisation
  - C = infinite: hard margin, needs to be clear space between the classes
  - If C is too big (or infinite), model may fail to converge
  - If C is too small, the model can underfit (learn a poor decision boundary)
- The kernel (which has it’s own parameters), which can be used to project the data into a higher dimensional space where it may be easier to separate
  - Linear: default, good first choice
  - RBF: can fit to complex distributions, gamma used to tune
  - Polynomial: controlled by the degree (or order) of the line, can be hard to tune
- Can use class weights to deal with class imbalance
  - Weight incorrect decisions in inverse proportion to the number of samples

K-Nearest Neighbours Classification (CKNN)

- K-Nearest Neighbours
  - Classification based on finding similar points
  - Sensitive to dataset size, and the number of neighbours
  - Compared to SVMs
    - Can more easily capture non-linear relationships via use of neighbours
    - Can be sensitive to outliers via use of neighbours
  - Extends to multi-class classification trivially
  - No actual learning
    - Decision boundary is based directly off provided input points

K-Nearest Neighbours Classification (CKNN) Parameters

- K, the number of neighbours
  - Bigger values of K will learn smoother boundaries, be less sensitive to noise, but will make rare classes harder to classify
- The distance metric
  - CKNN relies on finding the closest N points, changing the distance metric changes what we define as “close”
  - Euclidean is a good default if in doubt
- Distance weighting scheme
  - Default is uniform, all K neighbours are equal
  - Can use an inverse scheme, where closer neigbours are given a higher weight relative to their proximity
  - Inverse distance weighting can help when using a large K with rare classes present

Random Forests

- Ensemble of decision trees
  - Relies on classifiers being uncorrelated
  - Each classifier uses a random selection of training points
  - Each branch in each tree uses a random set of features
  - Inherently multi-class
  - Typically, very efficient
- Depth of tree leads to more accurate decisions on training data
  - If trees get too deep, can lead to overfitting
- Class weights can be included in the same manner as an SVM
- Can obtain a likelihood from results across all trees
  - i.e. 50% of trees said class 1, 25% said class 2, etc

Random Forest Hyperparameters

- Tree depth
  - Deeper trees allow for more fine-grained decisions, but increase overfitting
  - Generally, more classes and/or more dimensions will require deeper trees
- Number of trees
  - More trees means a larger cohort of models that are averaged
  - Small numbers of trees will lead to a greater sensitivity to noise
  - Run-time and memory will increase linearly with the number of trees, huge forests can become impractical

Selecting Hyper Parameters

- Changing hyper parameters can lead to a big change in performance
- If in doubt
  - Start with default values (they're defaults for a reason)
  - Use a grid-search
  - If run-time is a consideration
    - Possibly start with a coarse grid search, then refine around that
    - Run small scale tests to explore performance
    - Perhaps select 20% of the data, train some models, use this to inform decisions
  - Consider what performance metric you should be optimising for
    - F1? Accuracy?

Deep Learning

A SOMEWHAT POPULAR MACHINE LEARNING APPROACH

Deep Neural Networks

- Now state of the art for most machine learning tasks, but
  - Require lots of data
    - Or lots of tricks to work with limited data
  - Can be very resource intensive to train
- In general
  - Deep networks lead to greater representative power
  - But at very high depth training becomes difficult and architectural tweaks are needed
- Network architectures are flexible
  - The same network structure can be used for multiple problems
  - Tweak the input or output shapes, losses, etc as needed

Learning with Neural Networks

- Classification with deep neural networks
  - For a multi-class classification task
    - Output layer with N neurons, where N is the number of classes
    - Softmax activation, to emphasise 1 output above all others
    - Categorical cross entropy loss
  - For a binary classification task
    - Output layer with 1 neuron
    - Sigmoid activation, to push output towards either 0 or 1
    - Binary cross entropy loss
- Can use class weights to counter issues of class imbalance

Making Networks Better

- Overfitting can be an issue
- To address overfitting we may consider
  - Dropout
    - Though be careful when applying to convolutional layers
  - BatchNorm
    - Makes training easier by ensuring that values in the middle of the network are in a known range
  - Weight Regularisation
    - Implementation varies according to platform
  - Fine Tuning
    - Take an existing network, and adapt it to a new task
  - Data Augmentation
    - Create additional data by applying simple transforms to what you have
  - Fine-tuning and data augmentation can be used to help train networks with small datasets

Feature Representations

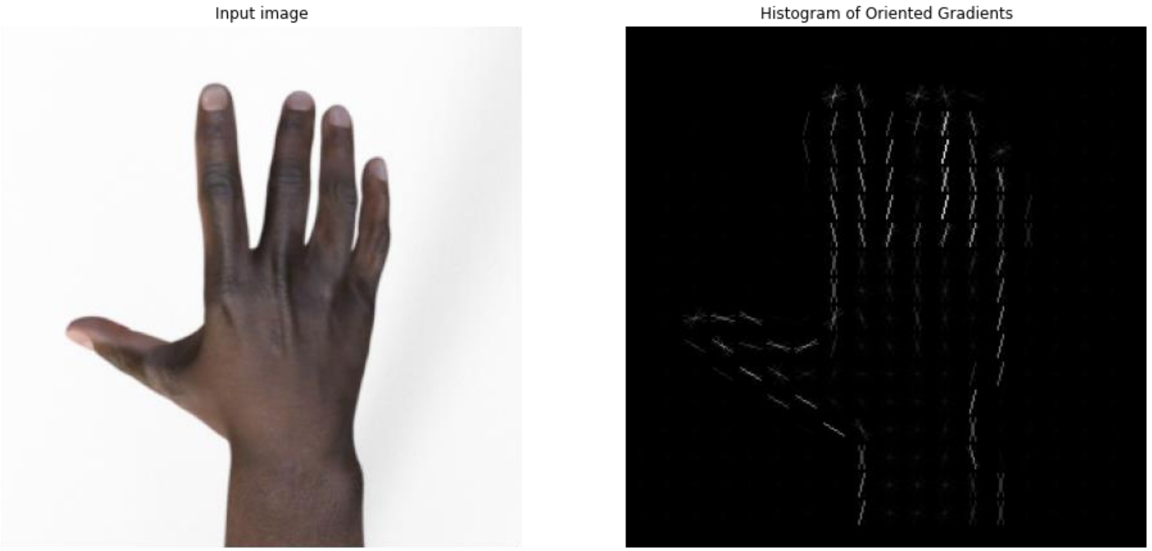
SOMETHING ELSE TO COMPLICATE THINGS

Feature Extraction

- So far, we’ve (mostly) used data “as is”
  - This is not always best
- Raw data may:
  - Contain useless or redundant information, or noise that hinders classification
  - Be very high dimensional
  - Not highlight the most important information for our task
- Feature extraction can be used to help
- Feature extraction will transform the data to another representation that is better suited to our task
  - Generally very domain specific, different methods exist for text, images, audio, etc
  - Methods typically have several parameters, which may need to be tuned
- A complete coverage of feature extraction is not possible in CAB420
  - We’ll look a couple of methods here for images and text

Histogram of Orientated Gradients (HOG)

- Extract features based on local texture
- The process is as follows:
  - Break image into patches
  - Compute gradients for each pixel in each patch
    - Compute magnitude and direction of gradient
    - High magnitude gradients will occur at the edge of the objects/regions
    - Direction of gradient indicates the direction of the edge
  - Quantise directions into a number of predefined directions
  - Build histograms for each region
  - Concatenate features for regions to obtain an overall image feature
- Image taken from ***CAB420_Classification_Bonus_Example_HOG.ipynb***

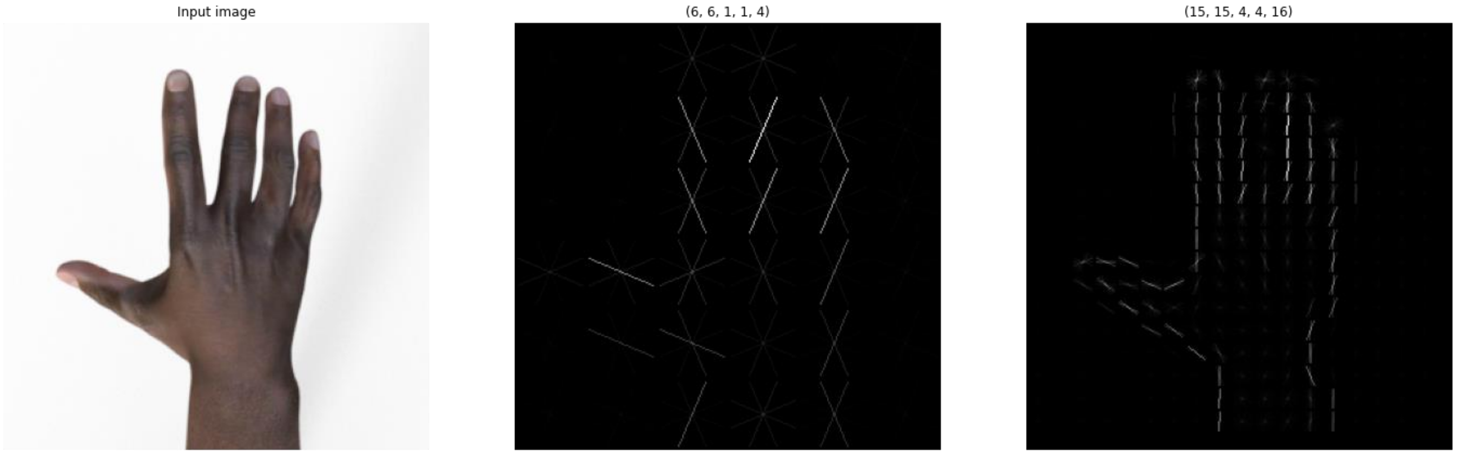


Input image

Histogram of Oriented Gradients

Histogram of Orientated Gradients (HOG)

- When extracting HOG you can select
  - Different numbers of regions/region sizes
  - Different numbers of gradient orientations
  - To aggregate features at a local region level
- Can lead to fine-grained or coarse features
- Image taken from ***CAB420_Classification_Bonus_Example_HOG.ipynb***

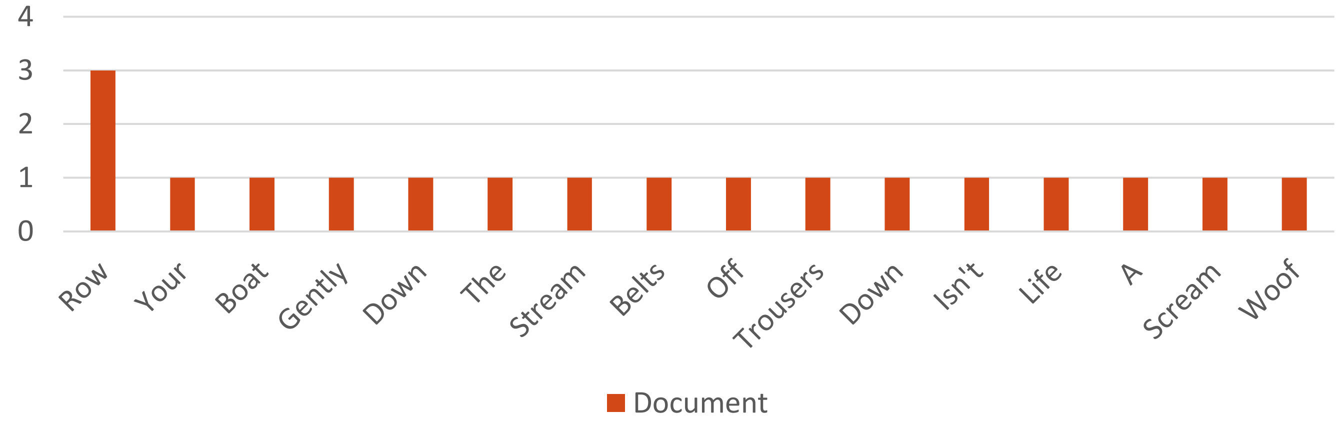


HOG vs Raw Pixles

- HOG is typically more compact (this does depend on HOG parameters)
  - Beneficial when dealing with small datasets
- HOG less sensitive to lighting changes
  - If the brightness of an image changes, the gradients don’t
- HOG throws away colour information
  - This may or may not be important for the problem
- HOG offers some small invariance to translation and rotation
  - How much depends on the size of regions being considered, and the number of gradient bins
- HOG parameters do need to be set
  - These are essentially extra hyper-parameters that need to be trained
- Multiple feature extractors can be used in combination
  - HOG + ???
  - Increase feature dimension
  - Increases feature discriminability

Bag of Words (BoW)

- Method to encode text data
  - Text samples are likely to vary in length, BoW encodes a sample into a fixed length
- Encode a document as a histogram that measures word occurrences
- Example document and histogram
  - “Row, row, row your boat, gently down the stream, belts off, trousers down, isn’t life a scream, woof!”



Bag of Words

- Transforms variable length data into a fixed length representation
- Destroys the order of that data
  - Hence the “bag”
- Histograms can be very large and sparse
  - There might be thousands of words, but documents may be quite small
  - Most documents will only contain a subset of all words
- Can tune the size of the dictionary
  - Can remove rare words
    - But rare words can be very informative
    - What’s rare?
  - Can remove really common words
    - If they’re really common, they’re probably not that informative

Bag of Words: Preparation

- We usually do some preprocessing before applying Bag of Words
  - Typically, very similar to what we do prior to learning embeddings
    - Convert to lowercase
    - Remove punctuation
    - Possibly remove plurals, or specific words
    - Tokenise document
      - Extract out each word on its own
- Can also build models based on combinations of words
  - You might use all words, and all pairs of consecutive words
  - Captures some order which is otherwise lost by BoW
  - Dictionaries get very big, very fast

Bag of Words

- Methods can be sensitive to the size of the document
  - Small documents will have small bags
  - Big documents will have big bags
- Can normalize based on document frequency
  - Often also use inverse frequencies
  - Term Frequency-Inverse Document Frequency (tf-idf) is a common weighting scheme

Feature Representations and Deep Learning

- Deep learning uses multiple layers to learn it’s own representations
  - Feature extraction is generally less common, or simpler, with deep learning
- We can think of early layers of a network as performing feature extraction
  - But the feature extraction is learnt on the data itself, so it’s tailored to the problem
- However, feature representations may still be with deep learning
  - For 1D data (audio, biomedical signals) frequency based transforms are common
  - For text data, text needs to be converted to a numeric form (typically word embeddings)
  - Data is often resized or resampled to a fixed size

Some Code Examples

FOR CLASSIFICATION

An Example

- See ***CAB420_Summary_2_Classification.ipynb***

- Classification of images (CIFAR-10), but using raw pixel values are features are not ideal
  - Pixel features are sensitive to changes in lighting, position, rotation, etc
  - Pixel features can be huge, a 200x200 colour image has 200x200x3 = 120,000 pixels, this is a lot of features
  - Feature extraction can be used to help
    - Histogram of Orientated Gradients

- We'll classify images using
  - SVM
  - CKNN
  - Random Forest
  - DCNN
    - We have spatial relationships in the data, so convolutions (2D being image data) make sense

- Review this example in your own time
  - Ask questions in class or via email/slack

An Example

- See ***CAB420_Summary_3_Text_Classification.ipynb***

- Classification of tweets into positive or negative sentiment
  - We'll use BoW features, though we could also use word embeddings
  - Word embeddings become less practical with longer sequences, though this is not such an issue with twitter data

- We'll classify tweets using
  - SVM
  - CKNN
  - Random Forest
  - DNN
    - Just a regular Deep Neural Network, no convolutions
    - Input data is a histogram, bins are not in any sort of order that makes sense for convolution operations

- Review this example in your own time
  - Ask questions in class or via email/slack