CS-618
Deep Learning
Late Spring 2023

Lecture 8

Anomaly Detection

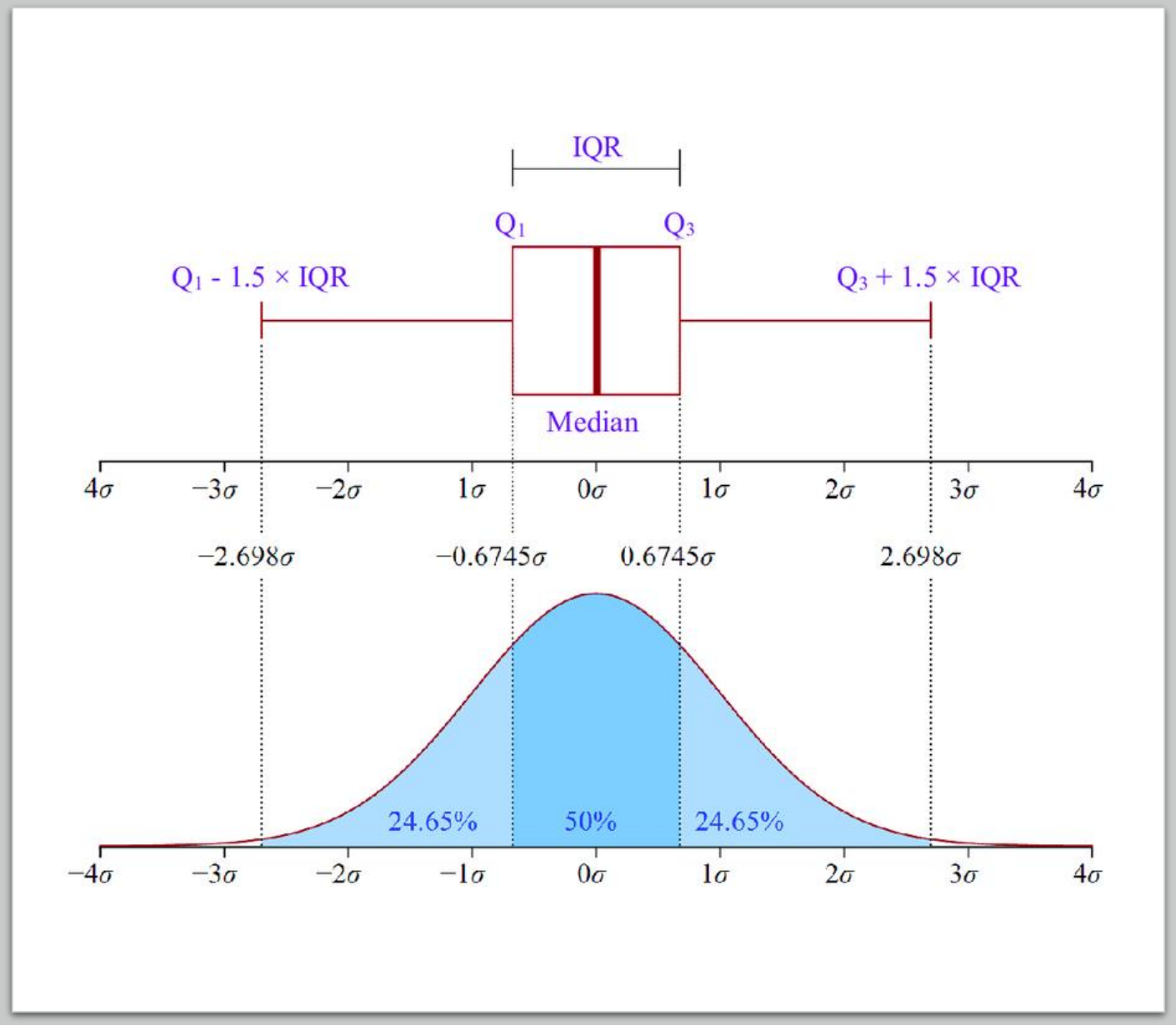
- Detecting abnormal data patterns or outliers is perfect use case for unsupervised learning
- Fraud detection, network intrusion detection, application errors, drop in traffic, increase in traffic, DDOS detection, abnormal login attempts, etc

Using stats to detect outliers

- Outlier – a data point that differs significantly from other observations
- Interquartile Range Outlier
- ZScores

Interquartile Range Outlier (IQR)

- First Quartile - number for which 25% of the data is less than that number
- Median – mid point
- Third Quartile – number for which 75% of the data is less than that number



IQR formula

- IQR = Q3 – Q1
- Outlier if X > (Q3 + (IQR * 1.5) or X < (Q1 – (IQR * 1.5)

Z-Scores

- Z-Score number of standard deviations from the mean. An outlier is considered if abs(Z-Score) > 3

$$Z = \frac{x - \mu}{\sigma}$$

Z = standard score

x = observed value

μ = mean of the sample

σ = standard deviation of the sample

NLP

- Natural Language Processing
  - Text Classification
  - Sentiment Analysis
  - Entity Detection
  - Chatbots

Natural Language Preprocessing Steps

- Corpus - Where words are gotten from (newpapers, db, websites, etc.)
- Tokenization – Splitting sentences into arrays of words
- N-Grams – building common phrases of words bigram and trigram most common
- Stemming – “Stemming is a technique used to extract the base form of the words by removing affixes from them. It is just like cutting down the branches of a tree to its stems. For example, the stem of the words ***eating, eats, eaten*** is ***eat***.
- Lemmatization – “Lemmatization technique is like stemming. The output we will get after lemmatization is called ‘lemma’, which is a root word rather than root stem, the output of stemming. After lemmatization, we will be getting a valid word that means the same thing”
- Bag of Words - words and # of occurrences
- Word Embeddings - Vectorizing words

Tokenization

- Removing punctuation
- Standardizing case
- Possibly removing numbers, emojis, stopwords, etc.

Stemming

- “Stemming is basically removing the suffix from a word and reduce it to its root word.
  For example: “**Flying**” is a word and its suffix is “**ing**”, if we remove “**ing**” from “**Flying**” then we will get base word or root word which is “**Fly**”.
- Over-stemming is when two words with different stems are stemmed to the same root. This is also known as a false positive.
  - Universal
  - University
  - universe
- Under-stemming is when two words that should be stemmed to the same root are not. This is also known as a false negative. Below is the example for the same.
  - alumnus
  - alumni
  - alumnae

Lemmatization

Lemmatization, on the other hand, takes into consideration the morphological analysis of the words. To do so, it is necessary to have detailed dictionaries which the algorithm can look through to link the form back to its lemma. Again, you can see how it works with the same example words.

Form	Morphological information	Lemma
studies	Third person, singular number, present tense of the verb study	study
studying	Gerund of the verb study	study
niñas	Feminine gender, plural number of the noun niño	niño
niñez	Singular number of the noun niñez	niñez

Stopwords

- “Stop words are words like “and”, “the”, “him”, which are presumed to be uninformative in representing the content of a text, and which may be removed to avoid them being construed as signal for prediction. Sometimes, however, similar words are useful for prediction, such as in classifying writing style or personality.”

TF_IDF

- Occurrence count is a good start but there is an issue: longer documents will have higher average count values than shorter documents, even though they might talk about the same topics.

- To avoid these potential discrepancies it suffices to divide the number of occurrences of each word in a document by the total number of words in the document: these new features are called tf for Term Frequencies.

- Another refinement on top of tf is to downscale weights for words that occur in many documents in the corpus and are therefore less informative than those that occur only in a smaller portion of the corpus

Word Embeddings

- A word embedding is a learned representation for text where words that have the same meaning have a similar representation.

Continuous Bag-of-Words (CBOW)

- The CBOW model learns the embedding by predicting the current word based on its context.

Continuous Skip-Gram Model

- The continuous skip-gram model learns by predicting the surrounding words given a current word.

NLP Libraries

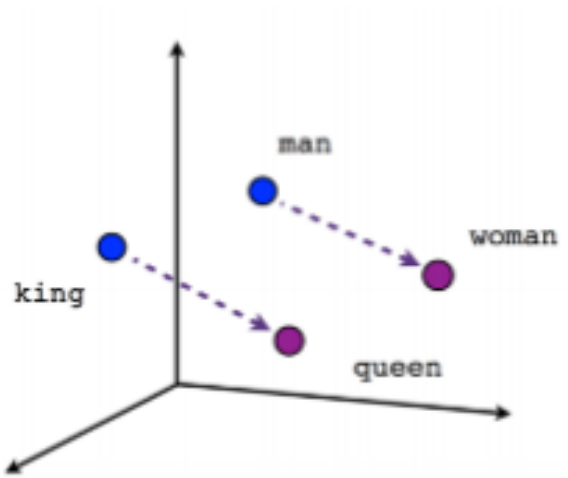
- NLTK https://www.nltk.org/
- Gensim - https://radimrehurek.com/gensim_3.8.3/
- https://spacy.io/
- https://fasttext.cc/docs/en/supervised-tutorial.html

NLP – Language Modeling

- Predict a word given a previous sequence of words
- For example, given the sentence, I went swimming in the ____, the model would predict the word “pool”
- language modelling can provision the much-needed linguistic knowledge (e.g. semantics, grammar, dependency parsing, etc.) to solve other downstream use cases like (information retrieval, question answering, machine translation, etc.).

Word2Vec and GloVe

The quest for learning language representations by pre-training models on large unlabelled text data started from word embeddings like Word2Vec and GloVe. These embeddings changed the way we performed NLP tasks. We now had embeddings that could capture contextual relationships among words.



Male-Female

Verb tense

Country-Capital

We have briefly discussed the task of language modelling. In a nutshell, language modelling, for the text w_1 $w_2,\ldots,w_{n-1},w_n$ where w_i is the ith word in the text, computes the probability of $w_n$given w_1 $w_2,\ldots,w_{n-1}$. In mathematical notation,

$$P(w_n|w_1, w_2, \ldots, w_{n-1})$$

In other words, it predicts w_n given $w_1,w_2,\ldots,w_{n-1}$ When training the model, we train the model to maximize this probability. i.e.

$$\text{argmax}_W P(w_n|w_1, w_2, \ldots, w_{n-1})$$

where the probability is computed using a model that has the trainable weights/parameters W. This computation becomes computationally infeasible for large texts as we need to look at from the current word all the way to the very first word. To make this computationally realizable, let's use the Markov property. Markov property states that you can approximate the above sequence with limited history. i.e.

LSTMs

- Input gate: Controls the amount of the current input that will contribute to the final output at a given time step
- Forget gate: Controls how much of the previous cell state affects the current cell state computation
- Output gate: Controls how much the current cell state contributes to the final output produced by the LSTM model

Gated Recurrent Units

- Implementation of LTSM without sacrificing performance
- Has 2 gates:
  - Update Gate - controls how much of previous layer is carried into current state
  - Reset Gate – how much of hidden state is reset with new input

Measuring Unsupervised Algorithms

- For example, if the language model is given the sentence "I like my pet dog" then ask to predict the missing word given "I like my pet _____", the model might predict "cat" and the accuracy would be zero. But that's not correct. "cat" makes as much sense as "dog" in this example. Is there a better solution here?
- Perplexity – measures how surprised the model was to see a target given the previous word sequence
- Entropy – measures the surprise, randomness, uncertainty of an event

The original interpretation of entropy is the expected value of the number of bits required to send a signal or a message informing of an event. A bit is a unit of memory which can be 1 or 0. For example, if you are the commander of an army who's at war with country A and B. Now you have 4 possibilities; A and B both surrender, A wins and B loses, A loses and B wins and both A and B win. If all these events are equally likely to happen you would need 2 bits to send a message, where each bit represents whether that country won or not. Entropy of a random variable X is quantified by the equation,

$$H(X) = -\sum_{x \forall X} p(x) log\big(p(x)\big)$$

where x is an outcome of X. Believe it or not, we have been using this equation without knowing it, every time we used the categorical cross-entropy loss. The crux of the categorical cross-entropy is this equation. Coming back to the perplexity measure perplexity is simply,

$$Perplexity = 2^{H(X)}$$

Predicting new text

- Greedy Decoding – predict one word at a time. (Given current word, predict the next word)
- Beam Search – predict several steps into future and selects the sequence with highest probability

BERT

- BERT (Bidirectional Encoder Representation from Transformers) is a type of transformer based model. *It is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context. As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of NLP tasks.*
- Transformer = attention mechanism that learns contextual relations between words in a text
- Allows model to learn context of a word based on all its surroundings (to the left and right of the reward)

Cloze Task

- Transformer models like BERT uses a variant of language modelling called “masked language modelling”. Masked language modelling is inspired by the Cloze task or the Cloze test. The idea is to, given a sentence with one or more blanks, ask the student to fill the blank. This has been used in language assessment tests to measure the linguistic competency of students. In masked language modelling the model becomes the student. Words are removed from inputs at random and the model is asked to predict the missing word. This forms the foundation of the training process used in models like BERT.

Transfer learning

- Transfer learning is a technique where a deep learning model trained on a large dataset is used to perform similar tasks on another dataset. We call such a deep learning model a pre-trained model

Data Drift

Concept drift or change in P(Y|X) is a shift in the actual relationship between the model inputs and the output. An example of concept drift is when macroeconomic factors make lending riskier, and there is a higher standard to be eligible for a loan. In this case, an income level that was earlier considered creditworthy is no longer creditworthy.

Prediction drift or change in P(Y hat Prediction|X) is a shift in the model's predictions. For example, a larger proportion of credit-worthy applications when your product was launched in a more affluent area. Your model still holds, but your business may be unprepared for this scenario

Label drift or change in P(Y Ground Truth) is a shift in the model's output or label distribution

Feature drift or change in P(X) is a shift in the model's input data distribution. For example, incomes of all applicants increase by 5%, but the economic fundamentals are the same.

Models In Production

- Incremental learning - Ability to retrain a model with fresher data vs. retraining from scratch
- Online/Active learning – learning in real-time in sequential order

Thank You