CMSC320 Exam 2: Study Guide

James Zhang*

May 15, 2024

Contents

*Email: jzhang72@terpmail.umd.edu

§1 Lecture 13: Introduction to Machine Learning

Definition 1.1. **Machine learning** is a subfield of AI that focuses on algorithms and statistical models to enable computer systmes to learn without being explicitly programmed. Decisions are based on patterns and insights derived from data. Traditional programs are explicitly programmed with rules.

Note 1.2.

- Supervised learning works with labeled data, used for classification and regression
- Unsupervised learning works with unlabeled data, trying to learn patterns or structures within the data, used for clustering or dimensionality reduction
- Reinforcement learning is when an agent learns in an environment to maximize some reward

Note 1.3. In supervised learning, the model is a learned fucntion mapping inputs to outputs. In unsupervised learning, learn patterns in input. In reinforcement learning, model-based or model-free methods (policy gradient).

§1.1 K-Nearest Neighbors

Definition 1.4. **K-Nearest Neighbors (KNN)** works based on the proximity between data points. Note that K is usually odd to avoid ties. Label the point based on the labels of the K closest points.

Note 1.5. Intuition behind **weighted KNN** is to give more weight to points nearby and less weight to points far away.

$$y = \sum_{k \in K} \frac{1}{\text{dist}(k, p)} * k_{\text{label}}$$

Definition 1.6. **Euclidean distance** is the most common distance metric, measures the straight line distance. Suitable for continuous numeric data.

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

Note 1.7. The other types of distance functions are

- **Manhattan (L1) distance**: $d = \sum_{i=1}^{n} |x_i - y_i|$. Good for high dimensional spaces, and cateogorical features since it ignores magnitude and only focuses on directional changes.
- **Hamming distance**: given two binary strings, count the number of different bits
- **Cosine Similarity**: Used for checking if vectors are pointing the same direction, used in text data, recommender systems

Definition 1.8. **Spherical K-Nearest Neighbors** is a variation of KNN specifically desgiend for datasets where the underlying data distribution is spherical. You can also predine sphere sizes, and then ket everything within the sphere vote $\implies$ faster at test time.

Note 1.9. How to choose K? Try multiple K's and graph the error rate, pick the lowest. This is known as **the elbow method**.

Note 1.10. If $K = 1$, then the models only considers the nearest neighbor when making predictions. Thus, the mdoel won't be able to generalize, and leads to overfitting. If $K = 1$, then we consider all points in the data, and we end up just choosing the mode $\implies$ underfitting.

Note 1.11. Strengths and weaknesses of KNN

- Strengths: accurate, few hyperparameters, easy to implement
- Weaknesses: no training step so it's a "lazy learner," slow at test time, does not work well with high dimensionality, sensitive to missing/noisy data, memory intensive, K must be chosen correctly

§2 Lecture 14: Feature Engineering and ML Evaluation

§2.1 Feature Engineering

Definition 2.1. **Feature Engineering** is when you transform raw data into features that are suitable for ML models. Selection involves choosing the most relevant features with most predictive power. Transformation (scaling, normalization, binning, mathematical transformations) to make features more suitable for transformations. Creation involves generating new faetures (combining, interaction terms, extracting other useful info).

Transformations (You need to memorize equations for below three for how to apply them using their equation)

Definition 2.2. **Normalization / Min-max Scaling** scales all values in a fixed range between 0 and 1.

$$x_{scaled} = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$$

Does not significantly alter distribution of the feature, and ensures no single feature dominates stats.

Definition 2.3. **Standardization (Z-score Normalization)** is

$$z = \frac{x_i - \mu}{\sigma}$$

such that the feature now has mean 0 and standard deviation 1. More robust to outliers, and in many cases, preferable to max-min normalization.

Definition 2.4. **Log Transforms** helps handle skewed data and after the transformation, the data becomes more like approximate to normal.

$$x = \log(x)$$

Definition 2.5. **One-Hot Encoding** is a data preprocessing technique that converts categorical variables into binary vectors. Uses pandas get dummies function.

§2.2 Cross Validation

Definition 2.6. **Training data** is the subset of the data used to train the data. **Testing data** is the subset of data set used to evaluate the model's performance. Train test split is risky if the split isn't random.

Definition 2.7. **Validation set** is a subset of the training data used iteratively for model development. The model is not trained on this data, helps make decisions about fine-tuning hyperparameters or model architecture.

Definition 2.8. **Cross validation** is a technique that makes sure the model gets to learn from different parts of the data in a more balanced way. The data is divided into subsets, called "folds". The model is trained on different folds, and one fold is used for validation set. Repeat the process using each fold as a validation set. Evaluates the model across all folds, which ensures that the model generalizes better to new data and avoid overfitting.

Note 2.9. There are different types of cross validation

- **K-Fold Cross Validation**: the process described above
- **Leave One Out Cross Validation (LOOCV)**: each data point is held out as the validation set while model trains on the rest of the data. Repeat this process for each data point. Useful if dataset is small but cen be computationally expensive for larger datasets.

§2.3 Evaluation Metrics

Note 2.10. Accuracy may not be able to show the full model performance, especially when classes are imbalanced. For example, a hihg accuracy might be achieved by always predicting the majority class, even if the model is performing poorly on the minority class.

Definition 2.11. Confusion Matrix: matrix that summarizes performance of a model on test data, shows true positives, true negatives, false positives, false negatives

- FP: model predicts positive, actually negative
- FN: model predicts negative, actually positive

Definition 2.12. The Confusion Matrix allows us to calculate various evaluation metrics.

- Sensitivity (recall) = $\frac{TP}{TP+FN}$. How many positves your model is able to recall from the data? The proportion of true positives to the total actual positives. Focuses on minimizing false positives in predictions.
- Precision = $\frac{TP}{TP+FP}$. Out of all predicted positives, how many are actually positive? Focuses on minimizing false negatives in predictions.
- Accuracy = $\frac{TP+TN}{TP+TN+FP+FN}$

Definition 2.13. F1-Score is used when you seek a balance between precision and recall, combines them into one score.

$$\text{F1 Score} = \frac{2}{\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Note 2.14. Precision vs. Recall:

- Testing for cancer $\implies$ recall; don't want to be wrong
- Legal system $\implies$ depends. Precision if the aim is to reduce fall accusations or arrests. Recall if priority is to capture all crimes.
- Fraud alerts $\implies$ recall, better to deny some good transactions than pay for the fraudulent ones
- Loans $\implies$ precision, only want to loan to people who will pay it back

§3 Lecture 15: Decision Trees

Definition 3.1. Decision Tree is a hierarchical data structure that represents data using a divide-and-conquer strategy. A decision tree is a set of questions that check if a condition is met and then flow from one into the other to make a decision.

Difference between root node, decision node, and terminal/child node

Definition 3.2.

- **Decision node**: where a specific feature of data is tested
- **Root node**: the first decision at the top of the tree
- **Branch/sub-tree**
- **Layer**
- **Leaf/Terminal node**: a node that does not test a feature and so is where a decision is made
- **Depth**: the number of layers in a decision tree

Note 3.3. Each internal node tests an attribute, and each branch corresponds to an attribute value. Each leaf node is assiged a classification. Also note that decision trees divide feature spaces into axis-parallel hyper-rectangles $\implies$ decision boundaries.

Note 3.4. Decision trees can represent any boolean function of the inputs. In the worst case, the tree will require exponentially many nodes.

Definition 3.5. **Pure node**: node wherein all datapoints belong to the same class

Note 3.6. Choosing the best attribute:

- Random: select any attribute at random
- Least-values: choose the attribute with the smallest number of possible values
- Most-values: choose the attribute with the largest number of values
- Max-gain: choose the attribute that has the largest expected information gain

Definition 3.7. **Impurity/entropy** measures the level of impurity in a group of examples. A split is a good split if we are more certain about classification after the split, so deterministic is good and uniform distributions are bad.

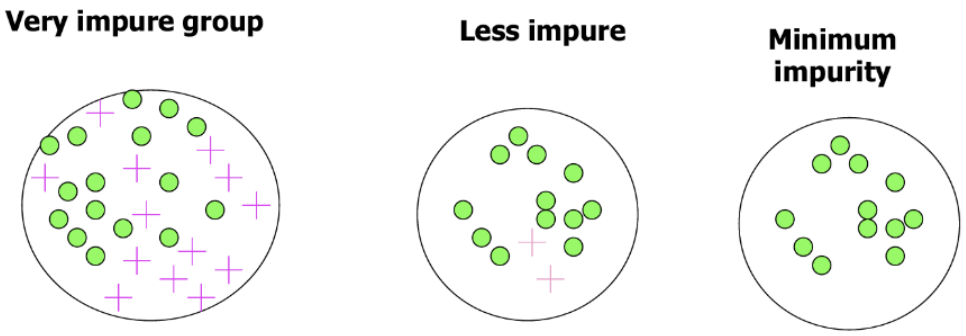
- When entropy $= 0$, the set is absolutely pure $\implies$ absolute certainty
- When entropy $= 1$, the state of absolute confusion

$$H(X) = -\sum_{i=1}^{n} \mathbb{P}(X = i) \log_2 \mathbb{P}(X = i)$$

From entropy to information gain, consider

$$\text{Gain}(s, a) = \text{Entropy}(p) - (\sum_{i=1}^{k} \frac{n_i}{n} \text{Entropy}(i))$$

where the parent node p is split into k partitions, and n_i is the number of records in partition i At each node of the decision tree, choose the attribute that maximizes information gain as the splitting criterion.



Note 3.8. How do you determine when to stop splitting?

- Reaching a maximum depth - prevents the model from becoming overly complex
- Having a minimum number of samples per leaf - splitting stops of a leaf node would contain fewer samples than a specified threshold
- Encountering pure nodes - nodes containing only one class

If a decision tree is given, for new test data predict the output/label using the given decision tree, slide 13
If a decision tree is given, what boolean function $(usingand(), or(V))$ can be represented, slides 15, 16
Don't forget to see the Cylinder example, where the boolean function for output GOOD is $cyl = 3Vcyl4etc$.

§4 Lecture 16: Classifications

§4.1 Naive Bayes

Definition 4.1. Naive Bayes is a classification algorithm that uses probability theory and Bayes' Theorem

$$\mathbb{P}(A|B) = \frac{\mathbb{P}(B|A) \times \mathbb{P}(A)}{\mathbb{P}(B)}$$

Example 4.2

Does "A very close game" belong to "Sports" or "Not sports"?

Solution. We want to compare $\mathbb{P}(\text{Sports} \text{ — a very close game})$ and $\mathbb{P}(\text{Not sports a very close game})$. Applying Bayes' Theorem, observe that

$$\mathbb{P}(\text{sports} \text{ — close game}) = \frac{\mathbb{P}(\text{close game} \text{ — sports}) \times \mathbb{P}(\text{sports})}{\mathbb{P}(\text{close game})}$$

$$\mathbb{P}(\text{not sports} \text{ — close game}) = \frac{\mathbb{P}(\text{close game} \text{ — not sports}) \times \mathbb{P}(\text{not sports})}{\mathbb{P}(\text{close game})}$$

If "a very close game" doesn't appear in training, this probability is 0. Naive Bayes' assumes that every word in a sentence is conditionally independent of other ones

$$\mathbb{P}(\text{a very close game}|\text{sports}) = \mathbb{P}(\text{a}|\text{sports}) * \mathbb{P}(\text{very}|\text{sports})$$

$$*\mathbb{P}(\text{close}|\text{sports}) * \mathbb{P}(\text{game}|\text{sports})$$

Instead of multiplying by 0 if at any time a word doesn't appear, add 1 to every count so it's never 0. This is called **Laplace Smoothing**.

$$\mathbb{P}(w_i|\text{class}) = \frac{\text{freq}(w_i, \text{class}) + 1}{N_{\text{class}} + V}$$

where N_{class} is the frequency of all words in the class and V is the number of unique words. □

§4.2 Support Vector Machines

Definition 4.3. **SVM** is an algorithm used for both regression and classification by defining the best decision boundary (hyperplane) that separates data points into different classes based on maximum margin.

Definition 4.4. A **hyperplane** is a flat surface that is one dimension lower than the input feature space.

Definition 4.5. Objective: find a plane that has **maximum margin**: maximum distance between data points of both classes $\implies$ larger buffer zone between categories

Definition 4.6. **Support vectors**: the closest data points to the hyperplane, has a big impact in deciding the hyperplane. Moving a support vector moves the decision boundary. Moving the other vectors has no effect.

Definition 4.7. **Kernel Trick** is used when data in the input feature space is not linearly separable. The Kernel Trick projects our data points into higher dimensional spaces by implicitly computing a dot product in higher dimensional space without explicitly transforming the data. In this higher dimensional space, the data could be

linearly separable. Popular kernels include polynomial kernel, Gaussian kernel, and sigmoid kernel.

Note 4.8. We do not need to know the SVM loss function.

§4.3 Overfitting and Underfitting

What is Overfitting and Underfitting

Definition 4.9. **Overfitting** is when the model becomes too tuned on the training set and is unable to generalize. This model is overly cokmplex and performs poorly on unseen data. Can be caused by noisy data, training set is too small, use KFCV or regularization techniques

Definition 4.10. **Underfitting** is when the model is too general, not complex enough. To do this, add more layers, more features, more epochs.

Know the concept of overfitting, and underfitting with respect to KNN, decision trees. K=1, K=N

Example 4.11

KNN with $K = 1$ overfits because highly sensitive to noise and outliers. KNN with $K = N$ underfits because it over generalizes and does not capture local patterns. Find optimal K using the elbow method.

Example 4.12

Decision trees have a big overfitting problem — they can create overly-complex trees that do not generalize to new data well. Apply a depth limit. Apply pruning where we just chop everything off after four levels. Or use ensembles of different trees $\implies$ random forests, which we cover next

§4.4 Random Forest

Definition 4.13. **Ensemble methods** such as bagging and boosting can also mitigate overfitting. These techniques combine multiple models to make predictions, reducing the impact of individual model's biases and errors. Enhance generalization and minimize overfitting.

Definition 4.14. **Random Forest** creates N different decision trees, each trained on a bootstrapped sample of training data generated randomly and with replacement.

§5 Lecture 17: Regressions

Definition 5.1. Classification predicts discrete labels, **regression** predicts a continuous feature.

Definition 5.2. The goal of **linear regression** is to predict output value using linear combination of input features. Find a function that represents the relationship between input variables (independent variables) and the output variable (dependent variable)

$$y = \beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n$$

Definition 5.3. **The line of best fit** is the one that minimizes error.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y - \beta_0 - \beta_1^T x)^2$$

$$RSS = \sum_{i=1}^{n}(y - \beta_0 - \beta_1^T x)^2$$

Definition 5.4. **Gradient Descent** is an iterative optimization algorithm to find the minimum of a function. In LR, can be used to minimuze RSS.

$$w_{new} \leftarrow \eta \frac{\partial J}{\partial w_{old}}$$

Definition 5.5. **Learning rate** determines the size of the steps taken in gradient descent. Big LR risks overshooting, small LR may converge slowly.

Evaluation methods R^2

Definition 5.6. R^2 is the coefficient of determination, a statistical method that determines the goodness of fit

$$R^2 = 1 - \frac{SSE}{SST}$$

Typically between 0 and 1 where 1 is a perfect fit and 0 means the model doesn't capture any variance.

Note 5.7. L.I.N.E, lasso, ridge, elastic net, not on exam.

§6 Lecture 18: Unsupervised Learning

What is unsupervised learning, when to apply it

Definition 6.1. **Unsupervised learning** analyzes and clusters unlabeled datasets.

Definition 6.2. **Clustering** groups data without the need for human intervention. Some applications are cluster analysis, dimensionality reduction, anomaly detection.

- Intra-cluster distances are minimized (similar objects should be put in the same cluster)
- Inter-cluster distances are maximimzed (observations in different clusters should be different)

§6.1 K-Means

Definition 6.3. **K-Means** is a clustering algorithm designed to create K clusters.

1. Specify the number of clusters k
2. Initialize the centroids
3. Assign each data point to the closest cluster center by calculating Euclidean distance and choosing the minimum.
4. Recompute centroids as cluster means

$$C(x, y) = ((x_1 + \cdots + x_n)/n, (y_1 + \cdots + y_n)/n)$$

5. Repeat 3 and 4 until the cluster centroids don't change

Note 6.4. If K is large, may defeat the point of clustering if clusters are so small. If $K = 1$ then one big cluster for the entire data set.

Note 6.5. **WCSS** is the sum of squares of the distances of each data point in all clusters to their respective centroids

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} ||x_i^{(j)} - c_k||^2$$

Variation within each cluster is minimized. As the value of K increases, the WCSS decreases rapidly. Choose the value of K at the elbow of this graph.

Note 6.6. The time complexity of K-Means is $\mathcal{O}(KNL)$ where K is the number of clusters, N is the number of examples, L is the number of iterations. K is a hyperparameter. Different initializations yield different results.

Note 6.7. The stopping criteria for K-Means is when the centroids of newly formed clusters do not change, not learning any new pattern. Points remain in the same cluster. Or max iterations reached.

§6.2 Principal Component Analysis

What is Dimensionality Reduction

Definition 6.8. **Dimensionality reduction** is the process of reducing the number of features in a dataset while preserving its essential information.

Definition 6.9. **Curse of dimensionality** is when the available data becomes sparser and fewer points relative to the size of the space. Increased computational complexity, risk of overfitting.

Definition 6.10. **PCA** attempts to account for the variance of data in as few dimensions as possible $\implies$ the new axes are principal components. The first PC is the axis that maximizes variance of the data. Second PC is orthogonal to the first and captures second highest variance after the first PC.

Note 6.11. Mean center the data. Compute the covariance matrix Σ. Eigendecompose the covariance matrix, and eigenvector with eigenvalue λ_1 is the first PC, etc

Note 6.12. To choose PCs, rank eigenvectors in order of eigenvalues. The kth eigenvalue captures the variance determined by its eigenvalue divided by the sum of all of the eigenvalues. Finally project the original data into the selected PC to obtain reduced-dimension representation of the dataset.

$$\text{FinalDataset} = \text{FeatureVector}^T * \text{StandardizedOrigDataset}^T$$

Note 6.13.

- Benefits: noise reduction, reduces number of features, indirectly helps feature selection
- Downsides: assumes linear relationships in data which isn't always true, PCs are hard to interpret, PCA doesn't generate new features

Note 6.14. Don't need to know how to perform PCA, calculate covariances, eigenvalues, or eigenvectors.

§7 Lecture 19: Neural Networks

What is logistic regression, when we use it

Definition 7.1. **Logistic Regression** is mainly used for binary classification. Note that it uses a **Sigmoid** layer as its last layer to squish output between 0 and 1

$$f(x) = \frac{1}{1 + e^{-x}}$$

What is a Neural Network, its different layers
Perceptron algorithm Forward Pass Calculations: How to calculate for each node using weight, input, bias, and apply an activation function Activation Function Sigmoid, how to apply it Backpropagation, no math needed What is multi-layer feedforward NN Fundamental basic mechanics

§8 Lecture 20: Image Processing

What is CNN
Difference between Traditional NN and CNN
Understand the basic mechanics
Know the different types of image classification tasks
Image Recognition Tasks
Calculations: Apply the convolution operation by multiplying the input image with the given kernel to create a 2D activation map.

Clearly show the steps involved in the convolution process, and provide the resulting activation map.
Check how stride values can affect the resulting output.
Calculations: Apply different pooling operations (max, average, sum poolings) to the given 2D activation map with the given filter and stride size.
You need to clearly show the steps.
Check how stride values can affect the resulting output.
Knowing what type of information people might lie about

§9 Natural Language Processing

Definition 9.1. **Syntax** is the structure of what makes the sentence valid, whereas **Semantics** is what the sentence actually means.

Note 9.2. Some of the following are problems in NLP: summarization, sentiment analysis, topic modeling, auto complete, question answering, translating.

Definition 9.3. **Sentiment analysis**: the process of classifying the emotional intent of text.

Definition 9.4. **Topic modeling** is an ML technique used to identify topics or themes present in a collection of text documents. Note that a document is a collection of words, and a corpus is a collection of documents.

Definition 9.5. Topic modeling is just supervised learning, but note that we have to make every word a vector. We have previously done so using one-hot encoding. The next natural approach is the **Bag of Words or Unigram Approach** - a representation of text that describes the occurrence of words within a document. This approach involves

1. A vocab of known words.
2. A measure of the presence of the words.

It's called a "bag" of words because order is discarded, and we only care about whether known words appear in the document. Create a table where columns are unique words and rows are document number, then the element is occurence.
Pros: easy to implement and Understand, efficient, applicable to any language
Cons: loss of word order, no context, vocab size could be large which could cause memory issues.

Definition 9.6. **N-Grams** is the more-sophisticated approach to creating vocab of grouped words. If we want more meaning, we can one-hot encode groups of words. **Bi-grams** is creating a vocabulary of two-word pairs. To get document vectors, have documents as rows, bi-grams as columns, and occurrences as elements of this matrix

Definition 9.7. First step of tokenization is to break text into chunks.

- Stemming - reducing words to thier root form (removes suffixes)

- Lemmatization - convert verb to infinitive form (running to run)
- Typo correction
- Stop words - the, and, etc

Definition 9.8. **TF-IDF (Term Frequency-Inverse Document Frequency)** is a way of measuring how relevant a word is to a document in a collection of documents.

- **Term Frequency (TF)** is how many times a term/word appears in a given document.
- **Document Frequency (DF)** - number of documents in which a word is present

We want to downweight words with high document frequency. The **IDF (the inverse document frequency)** of the word across a set of documents, such that rare words have high scores and common words have low scores.

$$TF - IDF = TF(t,d) \times IDF(t) = T(t,d) + (\log(\frac{N}{df(x)}))$$

the term frequency times the log of the number of docs / number of docs containing the word.
or way 2

$$IDF = \log \frac{1+N}{1+df_x} + 1$$

or way 3

$$IDF = \log \frac{1+N}{1+df_x}$$

Note 9.9. After done with vectorization with TF-IDF, apply cosine similarity.

Definition 9.10. **Cosine similarity** is a measure of the angles between two vectors, emphasizes direction not magnitude

$$\cos(\theta) = \frac{A \cdot B}{||A||\ ||B||} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

and note that this outputs values from -1 to 1. Smaller angles indicate greater similarity, even if different magnitude, the documents are probably the same topic. Different from other distance measures because magnitude is not considered, only direction.

§10 Graph Theory

Definition 10.1. A graph $G = (V, E)$ is a set of vertices V and edges E, where edges can be undirected or directed, unweighted or weighted

Definition 10.2. Basic graph vocab like degree, directed vs. undirected graph, weighted graph
In-degree is number of incoming edges to node, out-degree is number of outgoing edges to node

Definition 10.3. **Adjacency matrix** is a graph representation where rows and columns denote vertices. Also adjacency lists, adjacency dictionaries.

§10.1 Importance of Vertices

Definition 10.4. **Centrality analysis**: find out most important nodes in one network, there are 4 commonly-used measures

- Degree centrality - quantifies the number of connections a node has in a graph, $C_d(v_i) = \sum_j A_{ij}$
- Closeness centrality - importance here is measured by how close (central) a vertex is to other vertices. Look at average distances to all other nodes. Low average distance indicates higher centrality, more importance.

$$D_{avg}(v_i) = \frac{1}{n-1} \sum_{j \neq i}^{n} g(v_i, v_j)$$

$$C_c(v_i) = \frac{n-1}{\sum_{j \neq i}^{n} g(v_i, v_j)}$$

  where $g(\cdot)$ is the shortest path between two nodes

- Betweeness - counts the number of shortest oaths that pass thorugh the node, equation not needed. This identifies nodes that act as **bridges** along the shortest paths between pairs of nodes in a network

$$C_B(v_i) = \sum_{v_j \neq v_i \neq v_t} \frac{\sigma_{st}(v_i)}{\sigma_{st}}$$

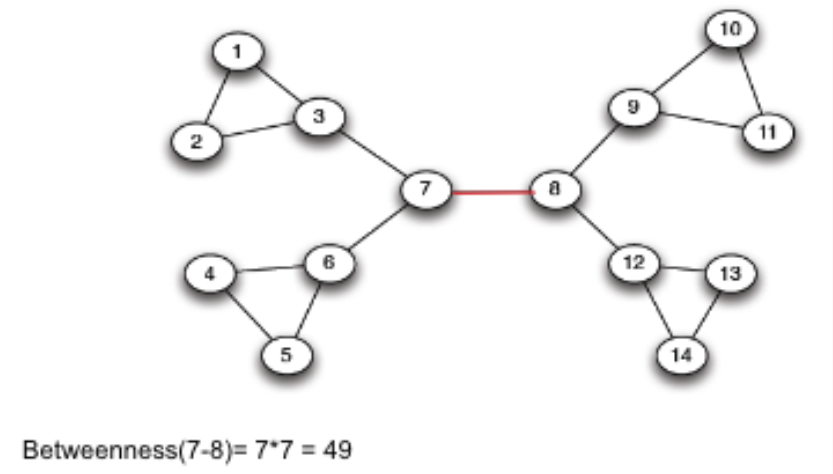
  where σ_{st} is the number of shortest paths between s and t, and $\sigma_{st}(v_i)$ is the number of shortest paths between s and t that pass through v_i.

Definition 10.5. **Bridge** connecting 2 different communities is a "weak tie". Bridges are rare in real networks. Relax the definition to be "an edge is bridge if the distance between 2 terminal veritices increases if the edge is removed".

Note 10.6. The larger the distance, the weaker the tie is. The larger the neighborhood overlap, the stronger the tie.

Definition 10.7. Nodes in a graph are probably in communities, which share common properties. Informally, a tightly-knit region of the network.

Example 10.8



The edge betweeness of the edge $7-8$ is 49 because there are 49 shortest paths that flow through this edge.

Definition 10.9. **Girvan-Newman Algorithm** is an algorithm for detection and analysis of community structure that relies on **iterative elimination of edges that have the highest number of shortest paths between nodes passing through them**. By removing edges one by one, we create communities.
Iteratively calculating edge betweeness for all edges, removing edge with highest edge betweeness, until all edges have edge betweenenss of 1
The method gives us only a succession of splits of the network into smaller and smaller communities, but doesn't tell us which split is best.

Definition 10.10. To find the best split, use **modularity**, measure of the strength of division of a network into modules or communities. **Modularity Score Q**. Identifying communties by maximizing modularity

§11 Recommendation Systems

Definition 11.1. **Recommendation Engines** are another common form of supervised learning, recommends the most relevant items to user

§11.1 Content Based Filtering

Definition 11.2. Content-based systems provide personalized recommendations to users. The main idea is that a user's preferences can be predicted based on previous interactions with items. Goal: recommend items to a user that are similar to items that they have previously interacted with. A content-based system works on similarities between products and so we need to

- Featurize each item, make a vector representing all product features
- Calculate similarities between vectors (euclidean, manhattan, jaccard, COSINE SIMILARITY)

- Build a model of user preferences, captures taste profile
- Identify potential recommendations
- Present curated recommendations

Pros: no need for other users, personalized, suggestions are not limited by popularity, easily understandable recommendations
Cons: feature selection challenging, building user profiles is otugh, content profile constraints, missed opportunities because can't recommend something the user has never encountered, periodic retraining needed

§11.2 Collaborative Filtering

Note 11.3. The main idea is that people who are similar may enjoy similar things. Furthermore, unlike content based, collaborative filtering does NOT require any information/features about the items

Definition 11.4. Collaborative filtering recommends users products on the basis of prefernces of the other users with similar tastes.

§11.2.1 User-Based Nearest Neighbor Collaborative Filtering

Definition 11.5. User-based, recommendations are based on preferences and behaviors of similar users.

1. User data collection - collect data on user preferences and interactions with the site
2. Create a user-item matrix
3. Similarity calculation, measure similarity between users based on their preferences
4. Neighborhood selection - identify subset of users (neighborhood) who are most similar to target user
5. Predict target user's rating for unseen items
6. Recommend the top N items

Definition 11.6. Jaccard Similarity: rating values do not matter, just count

$$\frac{|r_a \cap r_b|}{|r_s \cup r_b|}$$

problem is that rating values don't matter

Definition 11.7. Cosine similarity is back

$$sim(xx, y) = \frac{r_x \cdot r_y}{||r_x||||r_y||}$$

problem is that missing ratings are essentially negatives

Definition 11.8. Centered cosine: normalize ratings by centering row means, and then perform cosine similarity. Pearson correlation coefficient works well too.

Example 11.9

Let's work through an example. Here's a user-item matrix

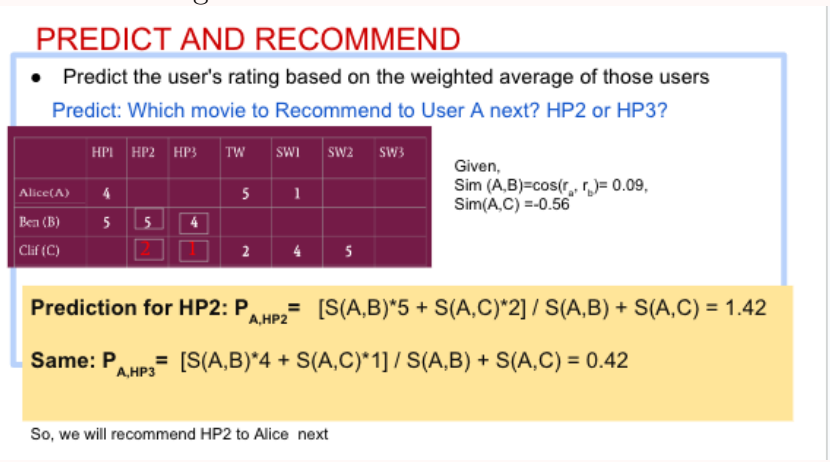
	HP1	HP2	HP3	TW	SW1	SW2	SW3
Alice	4			5	1		
Ben	5	5	4				
Clif				2	4	5	

Figure: User - Item Interaction Matrix where Movie Ratings is between [1-5]

From here, vectorize users by taking rows and the key idea: two users are similar if their vectors are similar! Usually do centered cosine similarity here.
Now KNN comes back. Given a user, find their k nearest neighbors. Predict the user's rating based on weighted average of those users.
For user x and product i, the predicted rating that x would give i is

$$r_{xi} = \frac{\sum_y s_{xy} r_{yi}}{\sum_y s_{xy}}$$

is the rate-weighted sum of similarities over sum of all user similarities.



§11.2.2 Item-Based Collaborative Filtering

Note 11.10. A user is likely to have the same opinion for similar items. Here, instead of focusing on similariteis between users, we focus on similarites between items

§11.3 Evaluations

§12 Association Rules

§13 Ethics