CS 270
Final Exam Review

Jake Cahoon

Table of Contents

Chapter 1

Introduction

1.1 Note From Jake

I hope the last review document was helpful, and I hope this one is as well. To make these, I've gone through the Exam Study Guide topics posted on Learning Suite. With the hope that I don't miss topics that will be on the final, I went through every slide and included all that I deem noteworthy. Despite this endeavor, I will miss topics, so I suggest you use this as a supplement to your study rather than relying solely on it.

I've included a table of contents this time around, so you can jump to the topics you need to study. That being said, let's friggin' do this.

Chapter 2

Data Mining Process Model

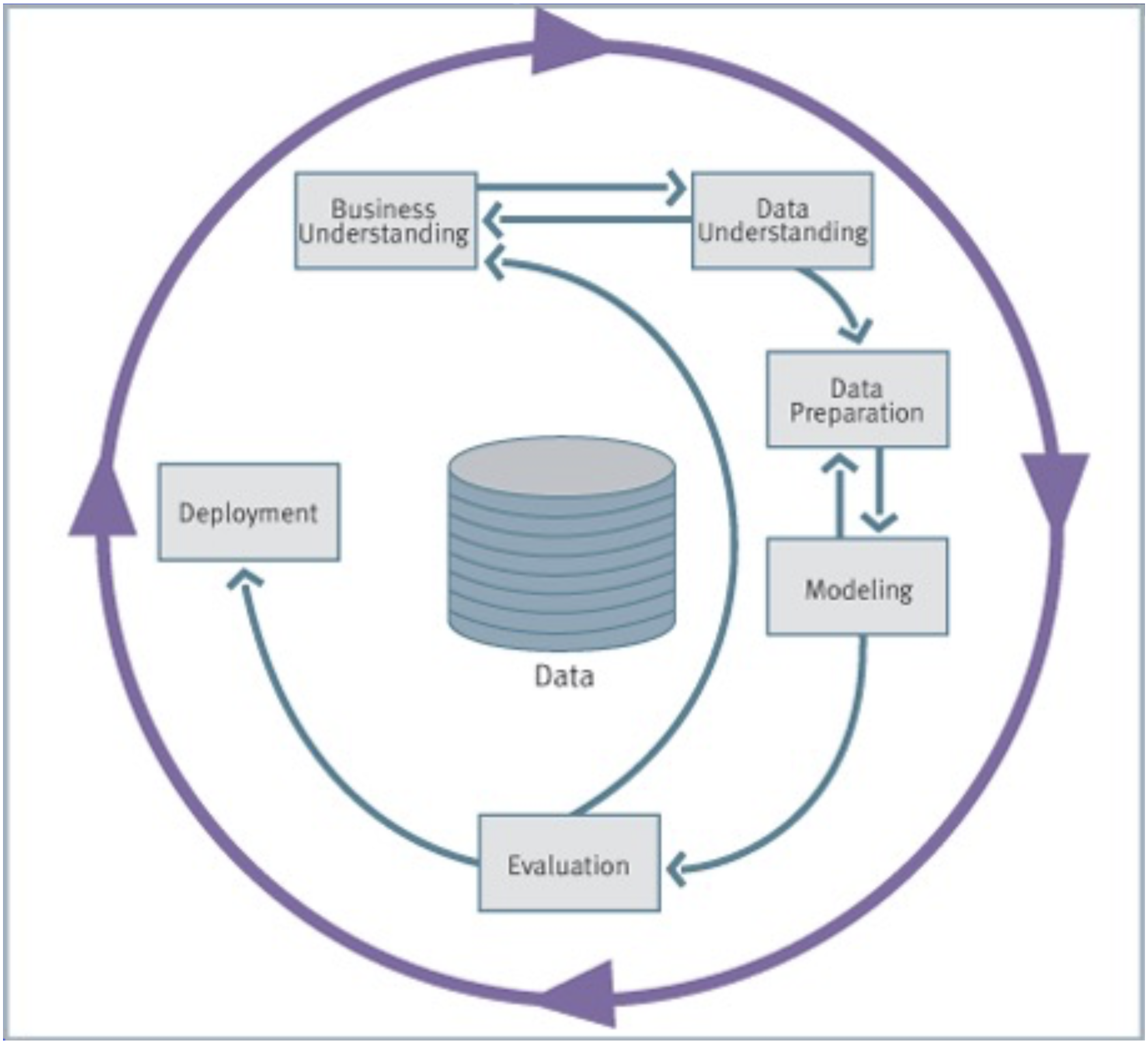
2.1 The Process

Note:-

I'm pulling straight from the slides here. Know this process at a high level.

1. Identify and define the task (business understanding)
   - Understand the context, audience, and problem
   - Tell the story
2. Gather and prepare the data
   - Build a dataset for the task
   - Select/transform/derive features
   - Conduct exploratory data analysis
   - Clean the data
3. Build and evaluate model
4. Deploy the model
   - Evaluate business related results
5. Iterate and improve the model

2.2 The Cycle Picture



Chapter 3

Bayesian Learning

There are two approaches to statistical learning: frequentist and Bayesian. Frequentist statistics is based on the idea of repeated sampling, while Bayesian statistics is based on the idea of starting with prior beliefs and then updating beliefs based on new information.

3.1 Bayes Theorem

Definition 3.1.1

Let C and A be events. Then

$$P(C|A) = \frac{P(A|C)P(C)}{P(A)}$$

Note:-

The right side of the equation is based on our current data, while the left side is what we want to find.

3.1.1 Notes

- Prior probabilities are based on prior knowledge. They are the initial beliefs.
- Posterior probabilities are the updated beliefs based on new information.
- $P(C)$ is the prior probability of the Class.
- $P(A)$ is the prior probability of the Attribute.
- $P(A|C)$ is the likelihood of the Attribute given the Class.
- $P(C|A)$ is the posterior probability of the Class given the Attribute.

3.2 Bayesian Classifiers

Given a set of attributes $\{A_1, A_2, \ldots, A_n\}$ and a class C, we can use Bayes Theorem to find the output class C that maximizes $P(C|A_1, A_2, \ldots, A_n)$. For each output class C, do

$$P(C|A_1, A_2, \ldots, A_n) = \frac{P(A_1, A_2, \ldots, A_n|C)P(C)}{P(A_1, A_2, \ldots, A_n)}$$

3.3 Maximum A Posteriori (MAP) Estimation

Definition 3.3.1

Let D be a dataset and let H be the set of all hypotheses. Then

$$\hat{h}_{MAP} = \operatorname*{argmax}_{h \in H} P(h|D)$$

where $\hat{h}_{MAP}$ is the maximum a posteriori hypothesis.

This is guaranteed to be "best", but it is computationally expensive and impractical for large hypothesis spaces.

3.4 Bayes Optimal Classifier

TODO: Figure out what to add here.

3.5 Naive Bayes Classifiers

A simple classifier that assumes that the attributes are conditionally independent given the class. This is a naive assumption, but it works well in practice. We assume that:

$$P(A_1, A_2, \ldots A_n|C) = P(A_1|C) \cdot P(A_2|C) \cdot \ldots \cdot P(A_n|C)$$

In other words, the probability of all the attributes given the class is the product of the probabilities of each attribute given the class. Then for each $A_i \in A$ and for each $C_j \in C$ we can estimate $P(A_i|C_j)$, which you did when you calculated the probabilities in the HW.

Once we have the probabilities, we can classify a new instance X by

$$\hat{C} = \operatorname*{argmax}_{C_j \in C} \left(P(C_j) \cdot \prod_{i=1}^{n} P(A_i|C_j) \right)$$

3.5.1 Notes

- Various $P(C_j)$ and $P(A_i|C_j)$ are estimated from the training data.
- Stores the probabilities in a table.
- For a new instance X, the classifier calculates the probability of each class given the attributes.
- $\hat{C}$ is the class with the highest probability.
- The true probability is the normalized probability.
- Independence assumption may not hold for some attributes.

Chapter 4

Ensembles

> Two heads are better than one, not because either is infallible, but because they are unlikely to go wrong in the same direction. - C.S. Lewis

4.1 Bias and Variance

- Bias is the error due to overly simplistic assumptions in the learning algorithm.
- Variance is the error due to the algorithm's sensitivity to fluctuations in the training data.

Bias and variance are inversely related. As one goes up, the other goes down. The goal is to minimize both.

4.2 Why are Ensembles Helpful?

By using ensembles, we can reduce the bias and variance of our models. Ensembles combine multiple models to create a stronger model. The idea is that the models will make different errors, and by combining them, we can reduce the overall error. See Dr. Snell's slides for examples.

4.2.1 Four Important Criteria

1. *Independence:* The models should be independent.
2. *Diversity:* The models should be different enough to make different errors.
3. *Decentralization:* The models should be trained on different subsets of the data.
4. *Aggregation:* The models should be combined in a way that reduces error.

4.3 Voting Ensemble

Let T be the training set, $A = \{A_1, A_2, \ldots, A_n\}$ be the set of models, and C be the set of classes. Define a function δ as follows:

$$\delta(a,b) = \begin{cases} 1 & \text{if } a = b \\ 0 & \text{otherwise} \end{cases}$$

Then the voting ensemble is defined as

1. For $k = 1$ to N, h_k is a model trained on T using learning algorithm A_k.
2. For a new instance X, the ensemble predicts

$$\hat{C} = \underset{c \in C}{\operatorname{argmax}} \sum_{k=1}^{N} \delta(c, h_k(X))$$

4.3.1 Notes

- Key issues: diversity and independence (too small of a set A and too similar models will not help).
- The models should be trained on different subsets of the data.

4.4 Bagging

Let T be the training set, A be the learning algorithm, N be the number of samples (or bags) of size d drawn from T, C be the set of classes, and δ be defined as in the Voting Ensembles section. Then the bagging ensemble is defined as

1. For $k = 1$ to N, $S_k \subseteq T$ is a sample of size d drawn from T, with replacement and h_k is a model trained on S_k using learning algorithm A.
2. For a new instance X, the ensemble predicts

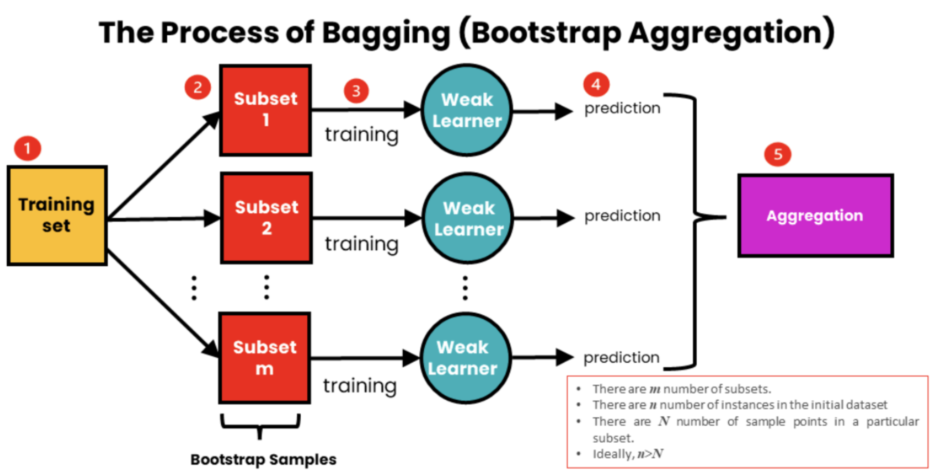
$$\hat{C} = \underset{c \in C}{\operatorname{argmax}} \sum_{k=1}^{N} \delta(c, h_k(X))$$

4.4.1 Random Forest

- A bagging ensemble extension of decision trees
- Each tree is trained on a different bootstrap sample
- Two random variables: the bootstrap sample and the feature subset

4.4.2 Notes

- Train N models on N different bootstrap samples
- Combines the outputs by voting (δ function)
- Decreases error by reducing variance due to unstable learning algorithms
- Homogeneous models are used

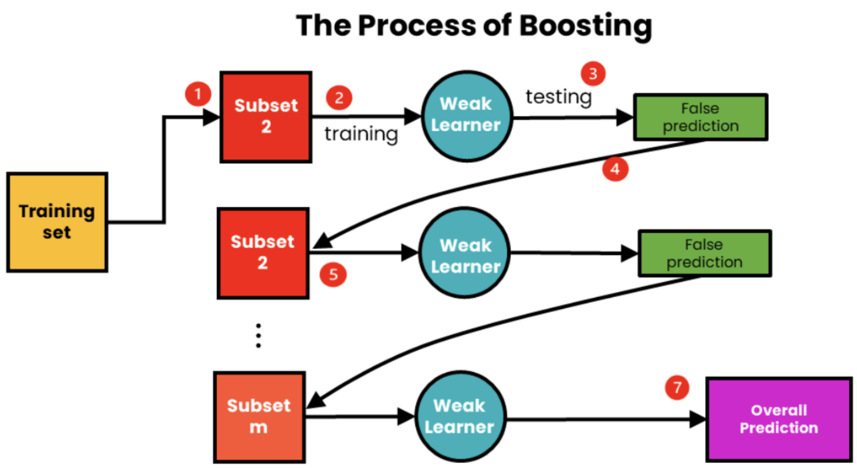


4.5 Boosting

"Boost weak learners into strong learners."

4.5.1 Notes

- Great for reducing bias
- Combines the outputs by weighted voting/averaging
- Homogeneous models are used
- Weak learners need to be better than random guessing



4.6 Stacking

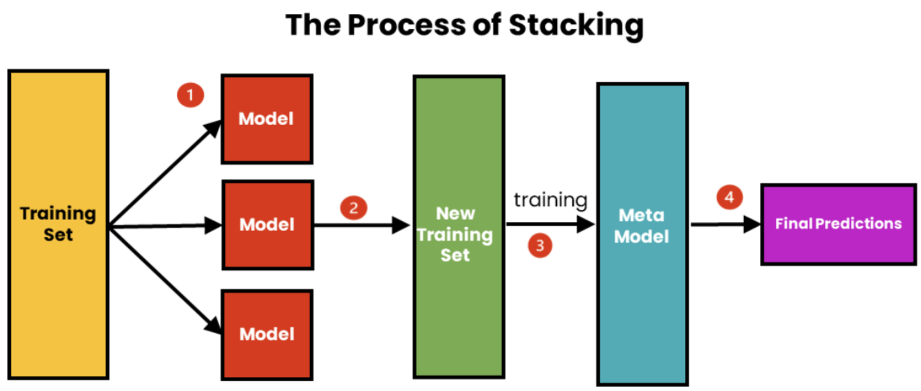
Let T be the base-level training set, N be the number of base-level learning algorithms, $A = \{A_1, A_2, \ldots, A_n, A_{\text{meta}}\}$ be the set of base-level learning algorithms, and A_{meta} be the chosen meta-level learner. Then the stacking ensemble is defined as

1. For $i = 1$ to N, h_i is a model trained on T using learning algorithm A_i.
2. Let T_m be the meta-level training set. $T_m = \emptyset$.
3. For $k = 1$ to $|T|$, $E_k = \{h_1(X_k), h_2(X_k), \ldots, h_N(X_k), y_k\}$.
4. $T_m = T_m \cup E_k$.
5. h_{meta} is a model trained on T_m using learning algorithm A_{meta}.
6. For a new instance X, the ensemble predicts

$$\hat{C} = h_{\text{meta}}(h_1(X), h_2(X), \ldots, h_N(X))$$

4.6.1 Notes

- Improves accuracy by combining the outputs of multiple models
- Hetereogeneous models are used at the base level
- The meta-level model is trained on the outputs of the base-level models


The Process of Stacking
Training Set
1
Model
Model
Model
2
New Training Set
training
3
Meta Model
4
Final Predictions

Chapter 5

Clustering

5.1 K-Means

5.2 Hierarchical Clustering

5.3 Silhouette Score

Chapter 6

Reinforcement Learning

6.1 Q-Learning

Chapter 7

Basic Precision/Recall

Chapter 8

CNNs

8.1 Structure

8.1.1 Convolutional Layers

8.1.2 Pooling Layers

8.1.3 Fully Connected Layers

Chapter 9

Other Deep Learning Topics

9.1 GANs

9.2 RNNs

9.3 LSTMs

9.4 Transformers