Announcements

• Project Milestone 3 due **Friday, December 9 at 8pm**
  • 2 day extension

• Final exam is **Thursday, December 22 from 6-8pm**
  • Review sessions today and Monday
  • Example final exam has been released
Agenda: Ethics

• Dataset issues

• Fairness/discrimination in ML models

• Misinformation about ML

• Feedback in ML systems

• Practical principles for ethical ML
Recap: Data Collection Issues

• Need to gather representative sample
• Need to ensure labels are unbiased
• Need to think carefully about whether to include sensitive attributes
Agenda: Ethics

• Dataset issues

• Fairness/discrimination in ML models

• Misinformation about ML

• Feedback in ML systems

• Practical principles for ethical ML
Recap: Group Fairness

<table>
<thead>
<tr>
<th>Non-discrimination criteria</th>
<th>Independence</th>
<th>Separation</th>
<th>Sufficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R \perp A$</td>
<td>$R \perp A \mid Y$</td>
<td>$Y \perp A \mid R$</td>
<td></td>
</tr>
</tbody>
</table>
Recap: Group Fairness

• **Independence:** Risk score distribution should be equal across ages:

\[ P(\text{risk score} \mid \text{age}) = P(\text{risk score}) \]

  • E.g., equal proportion of low risk customers for young vs. old people
  • Often called demographic parity

• If lower age groups in fact behave more riskily, algorithm is forced to make false negatives young people or false positives older people
Recap: Group Fairness

- **Separation:** Risk score should be independent of age given outcome:

\[ P(\text{risk score} \mid \text{age, true outcome}) = P(\text{risk score} \mid \text{true outcome}) \]

- Equivalent to saying the true positive rate and false positive rate are equal across subgroups

- **Example:** Both of the following hold:
  - Fraction of young, low-insurance-usage people correctly identified as low-risk
    \[ = \text{Fraction of old low-insurance-usage people correctly identified as low-risk} \]
  - Fraction of young high-insurance-usage people wrongly identified as low-risk
    \[ = \text{Fraction of old high-insurance-usage people wrongly identified as low-risk} \]
Agenda: Ethics

• Dataset issues

• Fairness/discrimination in ML models

• Misinformation about ML

• Feedback in ML systems

• Practical principles for ethical ML
Ethical Issues

• When you build ML models, you are responsible for how it is eventually deployed
  • Face classifier may be used by an authoritarian government to track people or target minority subgroups
  • Technology may be used in safety critical settings without sufficient validation
Best Practices for Ethical ML

• Human augmentation
• Bias evaluation
• Explainability and justification
• Displacement strategy
Human Augmentation

• Assess the impact of incorrect predictions and, when reasonable, design systems with human-in-the-loop review processes

• Especially important in domains with significant impact on human lives (e.g. justice, health, etc.)
  • All stakeholders’ values and perspectives should be accounted for during algorithm design
  • Domain experts as human-in-the-loop reviewers of ML decisions
Bias Evaluation

• **Use tools to understand bias in ML models**
  • No standard strategy, need to careful consider potential sources of bias for the domain you are working in
  • Requires continuous monitoring, not one-time effort
Explainability and Justification

• **Use tools to explain ML predictions**
  • Even though accuracy may decrease, the explainability may be significant
  • Important for end users to be able to understand ML predictions
  • Especially important due to hype and misinformation about ML

• **Challenges**
  • Potential leaking of sensitive data
  • Easy to game, e.g., “adversarial feedback”
  • Loss of competitive advantage
  • Sometimes hard to interpret, even for experts

Samuele Lo Piano, “Ethical Principles …” 2020
Explainability and Justification

• **Legal considerations**
  • France’s Digital Republic Act gives the right to an explanation as regards decisions on an individual made by algorithms
  • How and to what extent the algorithm was used, which data was processed and its source, etc.
  • Other countries considering similar laws
Displacement Strategy

• Identify and document relevant information so that business change processes can be developed to mitigate the impact on workers being automated

• Ensure all stakeholders are brought on board and develop a change-management strategy before automation

• Often, the workers are asked to do labor (e.g., generating training data) that will help automate themselves. Are the appropriately compensated?

Based on material from The Institute for Ethical AI and ML
Accountability

**Question:** Should a passenger in automated car be able to command it to go 80 MPH on a 55 MPH road?

**Reasons for “No”**
- It’s illegal and can endanger others
- Who is liable for accidents? Driver? Manufacturer? Insurance company?

**Reasons for “Yes”**
- Many exceptions!
- Rushing someone to the hospital, escaping a tornado, etc.
Other Challenges

• The ethics of ML and AI systems is an urgent topic now, not because of speculative future scenarios
  • Open and active area of research, involves scholars from law, social sciences, etc., as well as domain experts
  • Law moves slowly, and legal frameworks have much to catch up to

• Looking forward
  • **AI safety**: How can we make AI without unintended negative consequences?
  • **AI alignment**: How can AI make decisions that align with our values?
Useful Tools

• IBM AI Fairness 360: https://aif360.mybluemix.net/
• Google ML Fairness Gym: https://github.com/google/ml-fairness-gym
• Facebook Fairness Flow: https://venturebeat.com/2021/03/31/ai-experts-warn-facebooks-anti-bias-tool-is-completely-insufficient/
Concepts & Algorithms

• **Concepts**
  - Know these well
  - Especially bias-variance tradeoff!

• **Algorithms**
  - What does the model family look like?
  - What does the loss function measure?
  - How does the optimizer work?
  - What is the effect of each design decision and hyperparameter on bias-variance tradeoff and/or optimization?
Concept: Types of Learning

• **Supervised learning**
  • Predict unknown output given a new input
  • Most common task

• **Unsupervised learning**
  • Infer structure in unlabeled data
  • Automatically learn features, visualize data, etc.

• **Reinforcement learning**
  • Sequential decision-making in unknown environment
  • Robotics, control, etc.
Concept: Types of Learning

- Supervised Learning
  - Regression
  - Classification
  - Loss Minimization
  - Function Approximation
- Unsupervised Learning
- Reinforcement Learning
Concept: Loss Minimization View

• **Model family**: What are the candidate models $f$?

• **Loss function**: How to define “approximating”?

• **Optimizer**: How do we minimize the loss?
Algorithm: Linear Regression

• **Type:** Supervised learning

• **Model family:** Linear functions \( f_\beta(x) = \beta^T x \)

• **Loss function:** MSE \( L(\beta; Z) = \frac{1}{n} \sum_{i=1}^{n} (y_i - \beta^T x_i)^2 \)

• **Optimizer:** Gradient descent

• **Hyperparameters:** Learning rate \( \alpha \), convergence threshold \( \epsilon \)
Algorithm: Linear Regression

• Initialize $\beta_1 = \vec{0}$
• Repeat until $\|\beta_t - \beta_{t+1}\|_2 \leq \epsilon$
  \[
  \beta_{t+1} \leftarrow \beta_t - \alpha \cdot \nabla_\beta L(\beta_t; Z)
  \]
Algorithm: Linear Regression with Features

- **Type**: Supervised learning

- **Model family**: Linear functions $f_\beta(x) = \beta^T \phi(x)$

- **Loss function**: MSE $L(\beta; Z) = \frac{1}{n} \sum_{i=1}^{n} (y_i - \beta^T \phi(x_i))^2$

- **Optimizer**: Gradient descent

- **Hyperparameters**: Feature map $\phi$
Algorithm: Linear Regression with Features

• **Polynomial features**
  - $\phi(x) = \beta_1 + \beta_2 x_1 + \beta_3 x_2 + \beta_4 x_1^2 + \beta_5 x_1 x_2 + \beta_6 x_2^2 + \cdots$
  - Quadratic features are very common; capture “feature interactions”
  - Can use other nonlinearities (exponential, logarithm, square root, etc.)

• **Intercept term**
  - $\phi(x) = [1 \ x_1 \ \cdots \ x_d]^T$
  - Almost always used; captures constant effect

• **Encoding non-real inputs**
  - E.g., $x = \text{“the food was good”}$ and $y = 4$ stars
  - $\phi(x) = [1(\text{“good”} \in x) \ 1(\text{“bad”} \in x) \ \cdots]^T$
Concept: Bias-Variance Tradeoff

• **Overfitting (high variance)**
  - High capacity model capable of fitting complex data
  - Insufficient data to constrain it

• **Underfitting (high bias)**
  - Low capacity model that can only fit simple data
  - Sufficient data but poor fit
Concept: Bias-Variance Tradeoff

![Diagram showing the tradeoff between training and test loss]

- **Underfitting**: Low capacity, high bias, low variance
- **Ideal**: Balanced capacity, low bias, low variance
- **Overfitting**: High capacity, low bias, high variance

*Loss*  

*Capacity* (not actually 1D!)

*Training loss*   

*Test loss*

Slide by Padhraic Smyth, UCIrvine
Concept: Bias-Variance Tradeoff

![Graph showing the relationship between capacity and loss for overfitting and underfitting. The graph illustrates how training loss decreases as capacity increases, while test loss initially decreases but then increases with further increases in capacity. The ideal point is where both training and test loss are minimized.](image-url)
Algorithm: $L_2$ Regularized Linear Regression

- **Type:** Supervised learning

- **Model family:** Linear functions $f_\beta(x) = \beta^T x$

- **Loss function:** MSE $L(\beta; Z) = \frac{1}{n} \sum_{i=1}^{n} (y_i - \beta^T x_i)^2 + \lambda \cdot \|\beta\|_2^2$

- **Optimizer:** Gradient descent

- **Hyperparameters:** Regularization weight $\lambda$
Concept: Maximum Likelihood Estimation

• **Model family:** What is the likelihood $p(y | x)$?

• **Optimizer:** How do we minimize the negative log likelihood (NLL)?
Concept: Maximum Likelihood Estimation

• **Model family:** Most likely label

\[ f_\beta(x) = \arg \max_y p_\beta(y \mid x) \]

• **Loss function:** Negative log likelihood (NLL)

\[ \ell(\beta; Z) = -\sum_{i=1}^{n} \log p_\beta(y_i \mid x_i) \]
Algorithm: Linear Regression

- **Likelihood:** A Gaussian distribution

\[ p_\beta(y \mid x) = N(y; \beta^T x, 1) = \frac{1}{\sqrt{2\pi}} \cdot e^{-\frac{(\beta^T x - y)^2}{2}} \]

- **Optimizer:** Gradient descent
Algorithm: Linear Regression

• Model family:

\[ f_\beta(x) = \beta^T x \]

• Negative log likelihood:

\[ \ell(\beta; Z) = \frac{n \log(2\pi)}{2} + \sum_{i=1}^{n} (\beta^T x_i - y_i)^2 \]
Algorithm: Logistic Regression

- **Likelihood:** Bernoulli distribution with

\[
p_\beta(Y = 1 \mid x) = \frac{1}{1 + e^{-\beta^T x}} = \sigma(\beta^T x)
\]

\[
p_\beta(Y = 0 \mid x) = 1 - \sigma(\beta^T x)
\]

- **Optimizer:** Gradient descent
Algorithm: Logistic Regression

• Model family:

\[ f_\beta(x) = 1(\beta^T x \geq 0) \]

• Negative log likelihood:

\[
\ell(\beta; Z) = - \sum_{i=1}^{n} y_i \log(\sigma(\beta^T x_i)) + (1 - y_i) \log(1 - \sigma(\beta^T x_i))
\]
Concept: Regularization as a Prior

• What if we assume $\beta \sim N(0, \sigma^2 I)$?

• Consider the modified NLL

$$\ell(\beta; Z) = -\sum_{i=1}^{n} \log p_{\beta}(y_i \mid x_i) + \log \sigma \sqrt{2\pi} + \frac{\|\beta\|_2^2}{2\sigma^2}$$

constant  regularization

• Obtain $L_2$ regularization on $\beta$ with $\lambda = \frac{1}{2\sigma^2}$
Algorithm: Sensitivity vs. Specificity

<table>
<thead>
<tr>
<th>Actual Class</th>
<th>Predicted Class</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>TP</td>
<td></td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>FN</td>
<td></td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>FP</td>
<td></td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>TN</td>
<td></td>
</tr>
</tbody>
</table>

Sensitivity = \( \frac{TP}{TP + FN} \)

Specificity = \( \frac{TN}{TN + FP} \)
Algorithm: Sensitivity vs. Specificity

Each point on this curve corresponds to a choice of $\tau$

Aside: Area under ROC curve is another metric people consider when evaluating $\hat{\beta}(Z)$
Algorithm: KNN

- **Type**: Supervised

- **Model family**: Aggregate labels of $k$ nearest points
  - Parameters are the dataset ("nonparametric")

- **Loss function**: MSE, accuracy, etc.

- **Optimizer**: N/A

- **Hyperparameters**: Aggregation/distance functions, $k$
Algorithm: Decision Trees

• **Type:** Supervised

• **Model family:** Decision trees ($x_i = c$ for categorical, $x_i \leq t$ for real)

• **Loss function:** MSE, accuracy, etc.

• **Optimizer:** CART algorithm
  • Recursively choose nodes based on split that maximizes information gain
  • Early stopping (e.g., minimum gain) or prune using validation set

• **Hyperparameters:** Gain metric, maximum depth, minimum gain
Algorithm: Decision Trees
Concept: Ensemble Design Decisions

• How to learn the base models?
  • **Bagging**: Sub-sample dataset and features
  • **Boosting**: Iteratively upweight incorrectly classified examples
  • **Gradient boosting**: Train next base model on residual labels

• How to combine the learned base models?
  • Average, majority vote, etc.
  • Train a supervised learning model using base models as “features”
Algorithm: Random Forests

• **Type**: Supervised

• **Model family**: Average of decision trees
  • For classification, average predicted probabilities

• **Loss function**: MSE or accuracy

• “Optimizer”: Bagging
  • **Intuition**: Learn overfit decision trees and then “average away” variance

• **Hyperparameters**: Number of trees, bagging strategy, decision trees
Algorithm: Gradient Boosted Decision Trees

- **Type:** Supervised

- **Model family:** Weighted sum of decision trees

- **Loss function:** MSE or accuracy

- **“Optimizer”:** Boosting
  - **Intuition:** Train many shallow decision trees

- **Hyperparameters:** Number of trees, decision trees
Algorithm: Neural Networks

• **Type:** Supervised, unsupervised

• **Model family:** Custom composition of parametric layers
  • Nonlinearities
  • Linear/fully-connected, convolution, pooling, recurrent, self-attention

• **Loss function:** Any differentiable loss

• **Optimizer:** Gradient descent (compute gradient via backpropagation)
  • **Tweaks:** Momentum, adaptive learning rates, schedules, residual connections, initialization, batch normalization, dropout, early stopping
  • Make sure you know how to take partial derivatives!
Algorithm: Neural Networks

**AND**

- $x_1 + 20$
- $x_2 + 20$
- $x_1 - 30$

$(\neg x_1) \land (\neg x_2)$

**OR**

- $x_1 + 20$
- $x_2 + 20$
- $x_1 - 10$

NOT XOR

- $x_1 + 20$
- $x_2 + 20$
- $(\neg x_1) + 30$

$\neg \oplus$ in I

$\neg \oplus$ in III

Based on slide and example by Andrew Ng
Algorithm: K-Means Clustering

• **Type:** Unsupervised learning

• **Model family:** $K$ centroids, cluster is nearest centroid

• **Loss:** Average distance to nearest centroid

• **Optimizer:** Alternating minimization
  • **Step 1:** Given centroids, compute the cluster of each point
  • **Step 2:** Given clusters, compute the best centroids for each cluster

• **Hyperparameters:** Distance function, initialization strategy, $k$
Algorithm: PCA

- **Type**: Unsupervised learning

- **Model family**: K principal components (project point onto PCs)

- **Loss**: Approximation quality (e.g., MSE)

- **Optimizer**:
  - Center data and compute covariance matrix
  - Choose top $k$ eigenvectors with largest eigenvalues

- **Hyperparameters**: $k$
Algorithm: PCA
Algorithm: Word Vectors

• **Type:** Unsupervised (also called “self-supervised”)

• **Model family**
  - Neural network with a single hidden layer
  - Next word (and/or previous word)

• **Loss:** Softmax loss

• **Optimizer:** Gradient descent

• **Hyperparameters:** Hidden layer dimension
Algorithm: Word Vectors
Algorithm: Bayesian Networks

- **Type:** Supervised, unsupervised

- **Model family:** Parametric family of joint distributions $P(X_1, ..., X_k)$
  - Imposes constraints on structure of joint distribution
  - Need to perform inference to compute original joint distribution

- **Loss:** NLL: $- \sum_{i=1}^{n} \log P(X_1 = x_{1,1}, ..., X_k = x_{i,k})$

- **Optimizer:** Gradient descent

- **Hyperparameters:** Graph structure
Concept: MDP

- Set of states $s \in S$
- Set of actions $a \in A$
- Transition function $P(s' \mid s, a)$
- Reward function $R(s, a, s')$
- Discount factor $\gamma < 1$
Algorithm: Q Iteration

• **Type:** Reinforcement learning (to be precise, planning)

• **Model family:** Table of Q values $Q(s, a)$
  • Can use function approximation (use gradient update in optimizer)

• **Loss:** Cumulative expected reward

• **Optimizer:** Iteratively update Q values using Bellman equation

• **Hyperparameters:** Number of iterations, discount?
Algorithm: Q Learning

• **Type**: Reinforcement learning

• **Model family**: Table of Q values $Q(s, a)$
  • Can use function approximation (use gradient update in optimizer)

• **Loss**: Cumulative expected reward

• **Optimizer**: Iteratively update Q using approximate Bellman equation

• **Hyperparameters**: Learning rate, exploration strategy, discount?
Algorithm: Collaborative Filtering

• **Type:** Recommender system (between supervised and unsupervised)

• **Model family:** Predict ratings $X_{ik}$ of user $i$ for content $k$
  • Many choices, e.g., KNN using partial rating vectors

• **Loss:** MSE

• **Optimizer:** Model-dependent

• **Hyperparameters:** Distance/aggregation functions
Algorithm: Content-Based Recommendations

- **Type:** Recommender system (supervised)

- **Model family:** Predict ratings $X_{ik}$ of user $i$ for content $k$
  - Any supervised learning algorithm, e.g., linear regression

- **Loss:** MSE

- **Optimizer:** Model-dependent

- **Hyperparameters:** Item-content features
Concept: Ethics

• Dataset issues
• Fairness/discrimination in ML models
• Misinformation about ML
• Feedback in ML systems
• Practical principles for ethical ML
Concepts & Algorithms

• **Concepts**
  • Know these well
  • Especially bias-variance tradeoff!

• **Algorithms**
  • What does the model family look like?
  • What does the loss function measure?
  • How does the optimizer work?
  • What is the effect of each design decision and hyperparameter on bias-variance tradeoff and/or optimization?