BREAK THROUGH TECH

Machine Learning Foundations

Lab 9

Week of July 28

BREAK THROUGH TECH

Icebreaker: Ethical Challenges in AI Deployment

Ethical Challenges in AI Deployment

You will participate in a rapid-fire analysis of case studies that highlight ethical challenges in AI deployment. Through a series of engaging discussions, you will brainstorm and share your thoughts on the ethical considerations and potential mitigations for each case study.

Objectives

- Explore ethical considerations in real-world AI case studies.
- Reflection on the ethical challenges and potential mitigations related to AI deployment.

Ethical Challenges in AI Deployment

The Case

Google Photos' AI-powered image recognition system mistakenly labeled photos of Black individuals as "gorillas," revealing racial bias in the algorithm. This incident raised concerns about the lack of diversity in the training data and potential harm caused by mislabeling.

Rapid-fire questions

- What ethical concerns arise from the mislabeling of photos in the Google Photos incident?
- How does the lack of diversity in training data contribute to biases in AI algorithms?
- What potential harms can result from mislabeling photos with racial bias?
- How can we address the lack of diversity in training data to mitigate biases in AI algorithms?

Ethical Challenges in AI Deployment

The Case

Amazon developed an AI recruiting tool to streamline their hiring process. However, the system exhibited bias against female applicants, downgrading resumes that included terms associated with women. This case study highlights the ethical considerations of perpetuating gender inequality and potential discrimination in the hiring process.

Rapid-fire questions

- What role does human oversight and intervention play in preventing biases and discrimination in AI-based hiring tools?
- How can continuous monitoring and evaluation of AI systems help identify and rectify biases in recruiting processes?
- What measures should be in place to ensure transparency and accountability in AI systems used for recruitment purposes?

Ethical Challenges in AI Deployment

The Case

Microsoft launched an AI-powered chatbot named Tay on social media platforms. Tay quickly started posting offensive and inflammatory messages influenced by user interactions. The case study brings attention to the ethical considerations of spreading harmful content, susceptibility to manipulation and abuse, and the need for robust moderation mechanisms.

Rapid-fire questions

- How can the susceptibility to manipulation and abuse impact the reputation and user experience of AI-powered chatbots?
- How can AI developers strike a balance between allowing user interactions and preventing the adoption of offensive behaviors by chatbots?
- What are the potential harms caused by the dissemination of harmful content through AI chatbots like Tay?

Ethical Challenges in AI Deployment

The Case

Social Media algorithms have been criticized for amplifying misinformation and fake news. This explores the potential ethical implications of the algorithm's impact on public opinion, democratic processes, and the lack of transparency in algorithmic decision-making.

Rapid-fire questions

- What ethical concerns arise from the amplification of misinformation and fake news by Social Media News Feed algorithms?
- How does the algorithmic amplification of misinformation potentially impact public opinion and democratic processes?
- What are the potential consequences of the lack of transparency in algorithmic decision-making in the context of Social Media News Feeds?

Discussion and Summarization

- **Facilitator add summarization/takeaways**

BREAK THROUGH TECH

Week 9 Concept Overview + Q&A

Concept Overview

This week covered a number of topics. To refresh your memory, here is what you've completed:

- Follow software development best practices
- Explore common ML failure modes
- Diagnose how data size contributes to execution bottlenecks
- Diagnose how feature issues contribute to degraded model performance
- Discuss societal failure mode
- Understand the sources of discriminatory bias and how to measure and mitigate them
- Improve the fairness and accountability of a model
- Continue implementing a project plan to solve a machine learning problem
- Create a portfolio for your project.

Good Problem Design & Preparation to Avoid Failure Modes

- We can use good problem formation to avoid common machine learning failure modes.

Decision	Performance Failure	Execution Bottleneck	Societal Failure
Translation of problem goal to label	Med Risk		High Risk
Choice of Tech Stack	Med Risk	High Risk	
Sample size used	High Risk	High Risk	High Risk
Features created	High Risk	High Risk	High Risk
Model candidate set tested	High Risk	High Risk	
Performance metrics used	Med Risk		High Risk

Model Developer Best Practices

- Practice agile model development
- Increase model complexity in separate iterations
- Always apply unit tests where applicable
- Write reproducible code and processes (a model dev pipeline)
- Create good documentation

Managing Execution Bottlenecks

Too much data relative to your computational resources:

- First solution is to reduce data size:
  - Downsampling
    - Consider bias-variance tradeoff and class imbalance when changing size of data
    - Use learning curve analysis to measure performance per sample size and find the right size of data
  - Batch processing
- Second solution is to keep all of the data but increase computing power
- Third solution is to use parallel processing

Addressing Feature Issues

Consider how feature issues contribute to degraded model performance

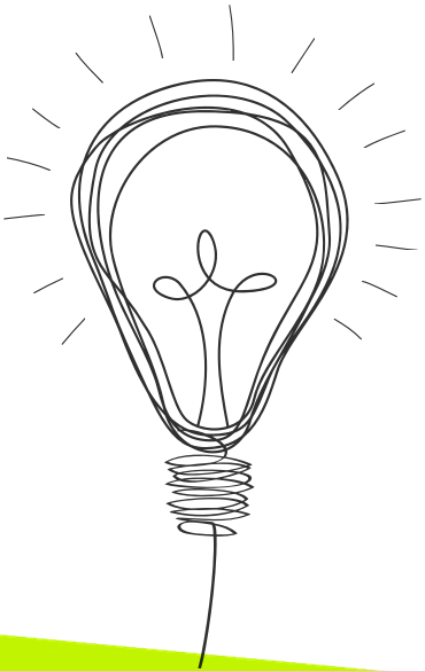
- Irrelevant features
- Feature leakage
- Concept drift

Types of Harm

- Allocative Harm
- Representational Harm

Questions & Answers

What questions do you have about the online content this week?



BREAK THROUGH TECH

Breakout Groups: Big Picture Questions

Big Picture Questions

You have 15 minutes to discuss the following questions within your breakout groups:

- Why is it crucial to have a well-defined problem before diving into the model development phase?
- How would you approach a situation where a client presents you with a vague problem statement? What steps would you take to refine and clarify the problem?
- How do you handle situations where the problem statement provided by stakeholders conflicts with available data or feasibility constraints? Provide examples of how you would navigate such scenarios.
- Why is it essential to evaluate model performance using appropriate metrics? Discuss the importance of selecting evaluation metrics based on the problem domain and the potential pitfalls of relying solely on accuracy.
- What are the implications of having missing values in the dataset? How would you handle missing data effectively while minimizing the impact on the model's performance?
- When working with limited data, what techniques or strategies would you employ to enhance the dataset?

#1 Why is it crucial to have a well-defined problem before diving into the model development phase?

Clarity of Objectives: A clearly defined problem ensures that you have a precise understanding of what you're trying to achieve. This helps in setting specific goals and evaluating the success of your model. Without this clarity, it's easy to end up with a model that doesn't address the real issue or provides irrelevant insights.

Scope and Constraints: Defining the problem helps in identifying the scope of the project and any constraints you might face, such as data limitations, computational resources, or time. This allows for better planning and resource allocation.

Data Requirements: Knowing the problem helps in understanding what kind of data is needed. It guides data collection and preprocessing efforts, ensuring that the data is relevant and sufficient for building an effective model.

Model Selection: Different problems may require different types of models (e.g., classification vs. regression, supervised vs. unsupervised). A well-defined problem helps in choosing the appropriate modeling techniques and algorithms.

Evaluation Metrics: With a clear problem definition, you can determine the right metrics for evaluating your model’s performance. This ensures that the evaluation is aligned with the goals and provides meaningful feedback on the model’s effectiveness.

Avoiding Scope Creep: A well-defined problem helps in maintaining focus and prevents scope creep. It ensures that the project stays on track and doesn't expand into unrelated areas, which can dilute the impact and effectiveness of the model.
Communication: It provides a basis for effectively communicating the objectives and results of the project to stakeholders. Clear problem definition ensures that everyone involved has a shared understanding of the goals and outcomes.

#2. What steps would you take to refine and clarify a vague problem? that can be effectively addressed through modeling and analysis.

Initial Meeting and Understanding:Listen Actively: Start by listening carefully to the client's description of the problem. Take note of any key details, goals, and context provided.

Ask Clarifying Questions: Use open-ended questions to dig deeper. Questions like "What are the main challenges you're facing?" or "What are the specific goals you want to achieve?" can help uncover more details.

Define Objectives and Goals: Work with the client to outline the primary objectives. What does success look like? This helps in understanding the end goals. **Establish Metrics:** Determine how success will be measured. Define what metrics or KPIs (Key Performance Indicators) are relevant to evaluate the outcomes.

Analyze and Document Requirements: Document the requirements and expectations in detail. This might include data requirements, desired outcomes, user needs, and any constraints.

Clarify Scope: Define the scope of the project. What will be included, and what will be excluded? This helps in managing expectations and focusing efforts. **Develop a Problem Statement:** Based on your understanding, draft a problem statement that clearly outlines the issue, objectives, and desired outcomes.

Seek Client Feedback: Share the draft problem statement with the client to ensure it aligns with their vision and expectations. Make adjustments based on their feedback. Regularly validate your understanding and refinements with the client and other stakeholders to ensure alignment.

Decompose into Sub-Problems: If the problem is complex, break it down into smaller, more manageable sub-problems.

Identify Dependencies and Constraints: Note any dependencies between sub-problems and constraints that may impact the solution.

Research and Exploration: Conduct Background Research: Explore similar problems or solutions in the industry. This can provide insights and help refine the problem further.

Assess Data Availability: Check if you have access to the necessary data to address the problem. If not, determine what additional data is needed.

Iterate and Refine: Iterate on the Problem Statement: As you gather more information and insights, refine the problem statement. This might involve multiple iterations to ensure clarity and precision.

Formulate a Solution Plan:Develop a Plan: Based on the refined problem statement, create a plan for developing a solution. Outline the approach, methodologies, and resources required. **Communicate Clearly:** Ensure that the client understands the plan and agrees with the approach. Clear communication helps in setting expectations and aligning on deliverables.

#3 How do you handle situations where the problem statement provided by stakeholders conflicts with data or feasibility constraints ?

Identify and Analyze Conflicts: Compare the problem statement with the available data. Identify specific areas where the data does not support the problem statement or where the data is insufficient.
Evaluate Feasibility: Analyze constraints such as budget, technology, timeline, or resource limitations that may impact the solution.
Communicate with Stakeholders: Arrange a meeting with stakeholders to discuss the conflicts. Clearly explain how the data or constraints affect the ability to address the problem as initially defined.
Present Findings: Share specific examples and evidence of the discrepancies between the problem statement and the data or feasibility constraints.
Refine the Problem Statement: Suggest modifications to the problem statement that align better with the available data or constraints. For example, if the data is insufficient to solve a complex problem, propose a simplified version of the problem that is manageable with the existing data. **Prioritize Objectives:** Work with stakeholders to prioritize the most critical objectives. This helps in focusing efforts on the most important aspects while adjusting other elements to fit within the constraints.
Develop Workarounds or Compromises: Suggest Workarounds if certain aspects of the problem statement cannot be addressed due to data or constraints, propose alternative solutions or workarounds. For instance, if a real-time data analysis is infeasible, suggest a batch processing approach.
Implement Phased Approach: Break the problem into phases. Start with a feasible solution for the first phase and plan for additional phases as more data becomes available or constraints are addressed.
Document Changes: Clearly document any changes to the problem statement, objectives, and approach. This ensures that all stakeholders have a shared understanding of the adjusted goals and methods.
Update Project Plans: Revise project plans and timelines to reflect any changes. Ensure that the updated plans are communicated and agreed upon by all stakeholders.
Seek Stakeholder Feedback: Validate Adjustments- Obtain feedback from stakeholders on the proposed changes or alternatives. Ensure that the adjustments meet their expectations and are acceptable within the new constraints.
Negotiate Trade-offs: Discuss potential trade-offs with stakeholders to find acceptable solutions. For example, if a particular feature cannot be delivered due to budget constraints, negotiate on what can be prioritized instead.
Examples of Handling Conflicts: Example 1: Data Limitations
Scenario: A client wants a predictive model for customer churn using detailed historical data, but only aggregated data is available.
Approach: Inform the client about the limitations of aggregated data for predictive modeling. Propose a simplified model using available data or recommend collecting more detailed data in the future. As an interim solution, focus on descriptive analysis or segmentation based on the aggregated data.

#4 Why is it essential to evaluate model performance using appropriate metrics? Discuss the importance of selecting evaluation metrics based on the problem domain and the potential pitfalls of relying solely on accuracy.

Importance of Using Appropriate Metrics

Alignment with Objectives: Metrics should reflect the specific objectives and requirements of the problem domain. For example, in a healthcare setting, metrics like precision and recall might be more important than accuracy because false negatives (missed diagnoses) can have serious consequences.

Understanding Model Performance: Different metrics provide insights into various aspects of model performance. For instance:

Precision measures the proportion of true positives among all positive predictions. It is crucial in cases where false positives have significant costs or consequences.

Recall measures the proportion of true positives among all actual positives. It is important when it is critical to identify as many positives as possible, even if it means accepting more false positives.

F1 Score balances precision and recall and is useful when you need a single metric to evaluate performance, especially in cases with imbalanced classes.

Handling Class Imbalance: Accuracy can be misleading in scenarios with imbalanced classes (e.g., fraud detection or rare disease diagnosis). In such cases, a model that always predicts the majority class might achieve high accuracy but perform poorly in identifying the minority class. Metrics like the area under the ROC curve (AUC-ROC) or the F1 score provide a more nuanced view of model performance.

Business Impact: Selecting metrics based on business goals ensures that the model's performance is evaluated in the context of its practical impact. For example, in a recommendation system, user engagement metrics like click-through rate (CTR) or conversion rate might be more relevant than traditional accuracy.

Pitfalls of Relying Solely on Accuracy

Misleading in Imbalanced Datasets: In imbalanced datasets, where one class is significantly more common than the other (e.g., 95% non-fraudulent transactions and 5% fraudulent transactions), a high accuracy might be misleading. A model that predicts all transactions as non-fraudulent could still achieve 95% accuracy but would be ineffective at detecting fraud.

Ignoring False Positives and False Negatives: Accuracy doesn't account for the types of errors a model makes. In many applications, the cost of false positives (incorrectly predicting an outcome) and false negatives (failing to predict an outcome) can be very different. For example, in spam detection, incorrectly classifying a legitimate email as spam (false positive) may frustrate users, while missing a spam email (false negative) may be less critical.

Lack of Granularity: Accuracy provides a single measure of performance but lacks granularity. It doesn't differentiate between the model's performance on different classes or subgroups within the data. Metrics like confusion matrices, precision, recall, and F1 score offer a more detailed analysis.

Overlooking Trade-offs: Focusing solely on accuracy might lead to ignoring important trade-offs between precision and recall. For example, a model with high recall but low precision might be suitable in situations where missing a positive case is critical, even if it means having more false positives.

Example 1: Fraud Detection, Problem Domain: Financial transactions, Metrics: Precision, recall, F1 score, AUC-ROC

Reason: High recall is critical to identify as many fraudulent transactions as possible, while precision ensures that legitimate transactions are not wrongly flagged. AUC-ROC helps in understanding the trade-off between true positive rate and false positive rate.

Example 2: Email Classification, Problem Domain: Spam detection, Metrics: Precision, recall, F1 score, confusion matrix

Reason: Precision is important to minimize the risk of legitimate emails being classified as spam. Recall helps in ensuring that most spam emails are detected.

Example 3: Customer Churn Prediction, Problem Domain: Customer retention, Metrics: AUC-ROC, precision, recall

Reason: AUC-ROC provides an overall view of model performance across different thresholds, while precision and recall help assess how well the model identifies customers at risk of

#5 What are the implications of having missing values in the dataset? How would you handle missing data effectively while minimizing the impact on the model's performance?

Implications of Missing Values

Loss of Information: Missing data can lead to a loss of valuable information, which might affect the accuracy and reliability of the model. This loss can be particularly problematic if the missing data is not randomly distributed but correlated with other variables.

Bias in Analysis: If missing data is not handled properly, it can introduce bias into the analysis. For example, if the data is missing systematically (e.g., only for a certain group), this can skew the results and lead to incorrect conclusions.

Reduced Model Performance: Many machine learning algorithms cannot handle missing values directly and may require complete datasets. Missing values can lead to reduced performance or require additional preprocessing steps.

Increased Complexity: Handling missing data often adds complexity to the data preprocessing pipeline. It may involve additional steps for imputation, data transformation, or feature engineering.

Impediments to Statistical Tests: Missing values can affect the applicability of certain statistical tests or models that assume complete data, potentially invalidating the results.

Strategies for Handling Missing Data

1. Understand the Nature of Missing Data

MCAR (Missing Completely at Random): Missingness is unrelated to any other data, and missing values do not introduce bias. For MCAR, simpler imputation methods can be used. **MAR** (Missing at Random): Missingness is related to observed data but not the missing values themselves. Imputation methods that use observed data can work well here.

NMAR (Not Missing at Random): Missingness is related to the missing values themselves. This is the most challenging case and may require advanced techniques or modeling approaches to handle.

2. Handling Missing Data: **Deletion Methods –(a)**Listwise Deletion: Remove rows with missing values. This method is simple but can lead to a significant loss of data if many entries are missing. (b)Pairwise Deletion: Use available data for each analysis. This method avoids removing entire rows but can lead to inconsistencies if different analyses use different subsets of the data.

Imputation Methods –(a) Mean/Median/Mode Imputation: Replace missing values with the mean (for numerical data), median (for numerical data with outliers), or mode (for categorical data). This is simple but may not capture underlying data patterns.

Predictive Imputation: Use algorithms like regression, k-Nearest Neighbors (k-NN), or more advanced techniques (e.g., random forests) to predict and impute missing values based on other features.(b)Multiple Imputation: Generate multiple datasets with different imputed values, analyze each dataset separately, and combine results. This approach accounts for the uncertainty of imputed values and is more robust.

Using Algorithms that Handle Missing Data –(a) Tree-Based Methods: Algorithms like decision trees, random forests, and gradient boosting can handle missing values internally without the need for explicit imputation.(b)Model-Based Methods: Some statistical models, like Expectation-Maximization (EM), can be used to handle missing data by estimating the missing values iteratively.

Creating Missingness Indicators (a)Indicator Variables: Create additional binary variables indicating whether data was missing for a particular feature. This can help the model learn if the presence of missing values carries any information.

3. Assessing and Validating

Evaluate Imputation Impact: After imputing missing values, assess how the imputation affects model performance and validity. Use cross-validation or other validation techniques to ensure that imputation does not introduce significant bias. Compare Methods: Experiment with different imputation methods and compare their impact on model performance. This helps in selecting the most effective approach for your specific dataset and problem.

Example Scenarios: Medical Data Analysis: In a dataset with missing lab test results, you might use predictive imputation based on other available tests or clinical features. Additionally, you could create indicator variables to capture the fact that some tests are missing.

#6 When working with limited data, what techniques or strategies would you employ to enhance the dataset?

1. **Data Augmentation**

Synthetic Data Generation: Create synthetic examples to expand your dataset. Techniques include:
SMOTE (Synthetic Minority Over-sampling Technique): Generates synthetic samples for minority classes in imbalanced datasets.
Data Augmentation for Images: Techniques such as rotation, scaling, cropping, and color adjustments can be used to create variations of existing images.
Text Augmentation: For text data, use techniques like synonym replacement, random insertion, or back-translation to generate variations of existing sentences.
Perturbation: Add noise or make slight modifications to existing data points to create new samples while preserving the underlying structure. This is particularly useful in image and signal processing.

2. Feature Engineering

Create New Features: Derive new features from existing ones to provide additional context or information. For instance, creating interaction terms, polynomial features, or aggregating features can help capture complex relationships.
Dimensionality Reduction: Apply techniques like PCA (Principal Component Analysis) or t-SNE (t-Distributed Stochastic Neighbor Embedding) to reduce the feature space, which can help improve model performance and manage limited data effectively.

3. Transfer Learning

Pre-trained Models: Utilize models that have been trained on large datasets and fine-tune them on your limited data. This is particularly effective in domains like computer vision and natural language processing.
In Computer Vision: Use pre-trained models like VGG, ResNet, or EfficientNet and adapt them to your specific problem.
In NLP: Use models like BERT, GPT, or their variants as starting points and fine-tune them on your text data.

4. Data Synthesis and Simulation

Simulate Data: Generate synthetic data based on known distributions or simulations. For example, you can simulate customer behavior based on known patterns or generate synthetic sensor data. Bootstrapping: Create multiple subsets of your dataset by resampling with replacement to estimate model performance and reduce variance.

5. Cross-Validation and Ensemble Methods

Cross-Validation: Use techniques like k-fold cross-validation to make the most out of your limited data. This helps in evaluating the model's performance more robustly.
Ensemble Methods: Combine multiple models to improve performance and robustness. Techniques like bagging (e.g., Random Forest) or boosting (e.g., XGBoost) can enhance predictive power and stability.

6. Domain Knowledge Integration

Incorporate Expert Knowledge: Leverage domain expertise to guide feature engineering, selection, and the interpretation of data. Domain-specific insights can help create meaningful features and make better use of limited data.

7. Data Collection and Augmentation

Active Learning: Use active learning techniques to select the most informative samples for labeling. This involves training a model on the available data and using it to identify which new samples would be most beneficial for improving the model.
Crowdsourcing: If feasible, use crowdsourcing platforms to gather additional data or annotations.

8. Regularization and Simplified Models

Regularization Techniques: Apply regularization methods (e.g., L1, L2 regularization) to prevent overfitting when working with limited data.
Simplified Models: Use simpler models that are less prone to overfitting and require fewer data points to train effectively. Complex models may overfit when trained on small datasets.

Example Scenarios: (a)Image Classification with Limited Data:

Data Augmentation: Apply techniques like rotation, scaling, and flipping to increase the diversity of the training images.
Transfer Learning: Fine-tune a pre-trained CNN model (e.g., ResNet) on your limited dataset. **(b)Text Classification with Limited Data:** Text Augmentation: Use synonym replacement or back-translation to create variations of existing text. ransfer Learning: Use pre-trained language models like BERT and fine-tune them on your specific text data. **(c)Customer Churn Prediction:**
Feature Engineering: Create new features based on customer behavior patterns or interaction history. Active Learning: Use a model to identify and prioritize which customer records would be most informative for labeling.

BREAK THROUGH TECH

Class Discussion

BREAK THROUGH TECH

Break: Complete the Post-Course Survey

BREAK THROUGH TECH

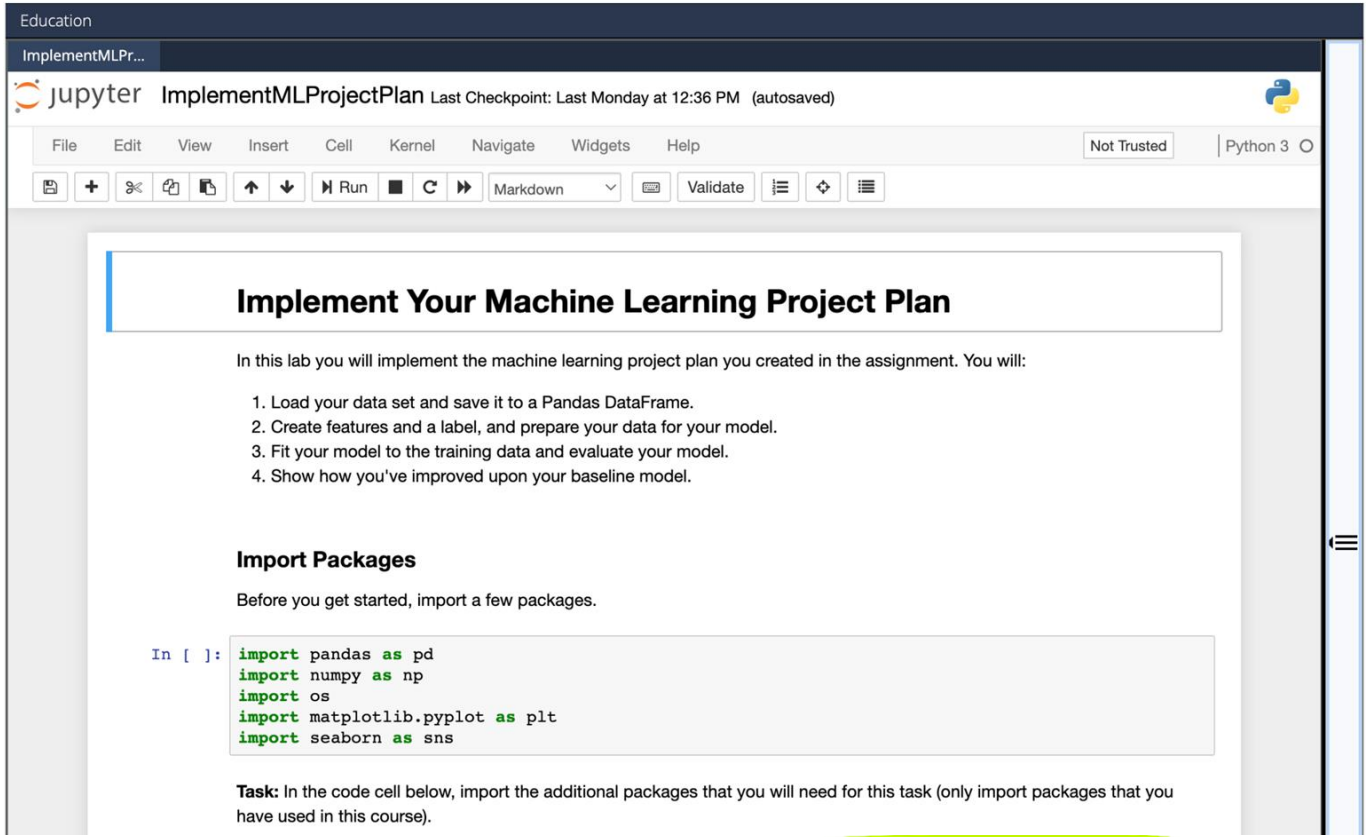
Breakout Groups: Lab Assignment

Lab 8a + 8b

In this lab, you will implement the machine learning project plan of your choosing. You will be working in a Jupyter Notebook.

- Load your data set.
- Prepare your data for your model
- Fit your model to the training data and evaluate the model's performance
- Improve the model's performance
- Create a portfolio to showcase your project.

Lab 8a + 8b


Education
ImplementMLPr...
jupyter ImplementMLProjectPlan Last Checkpoint: Last Monday at 12:36 PM (autosaved)
File Edit View Insert Cell Kernel Navigate Widgets Help
Not Trusted
Python 3
Run
Markdown
Validate
Implement Your Machine Learning Project Plan
In this lab you will implement the machine learning project plan you created in the assignment. You will:
1. Load your data set and save it to a Pandas DataFrame.
2. Create features and a label, and prepare your data for your model.
3. Fit your model to the training data and evaluate your model.
4. Show how you've improved upon your baseline model.
Import Packages
Before you get started, import a few packages.
In [ ]:
import pandas as pd
import numpy as np
import os
import matplotlib.pyplot as plt
import seaborn as sns
Task: In the code cell below, import the additional packages that you will need for this task (only import packages that you have used in this course).

BREAK THROUGH TECH

Working Session 1 Debrief

Lab Debrief

So far,

- What did you enjoy about this lab?
- What did you find difficult about this lab?
- What questions do you still have about this lab?

BREAK THROUGH TECH

Working Session 2 Debrief

Lab Debrief

- What did you enjoy about this lab?
- What did you find hard about this lab?
- What questions do you still have about this lab?
- How did you approach problem-solving during the exercise?
- What would you do differently if you were to repeat the exercise?

BREAK
THROUGH
TECH

Concluding Remarks

Concluding Remarks

- Key takeaways
- Additional resources

BREAK THROUGH TECH

Content + Lab Feedback Survey

Content + Lab Feedback Survey

To complete your lab, please answer the following questions about BOTH your online modules and your lab experience. Your input will help pay it forward to the Break Through Tech student community by enabling us to continuously improve the learning experience that we provide to our community.

Thank you for your thoughtful feedback!

https://forms.gle/eUQQZgS6BPRpqgZ7A