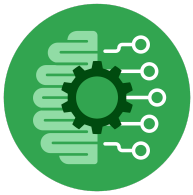
Course Six
The Nuts and Bolts of Machine Learning



Instructions

Use this PACE strategy document to record decisions and reflections as you work through the end-of-course project. As a reminder, this document is a resource that you can reference in the future and a guide to help consider responses and reflections posed at various points throughout projects.

Course Project Recap

Regardless of which track you have chosen to complete, your goals for this project are:

- ☑ ~~Complete the questions in the Course 6 PACE strategy document~~
- ☑ ~~Answer the questions in the Jupyter notebook project file~~
- ☑ ~~Build a machine learning model~~
- ☑ ~~Create an executive summary for team members and other stakeholders~~

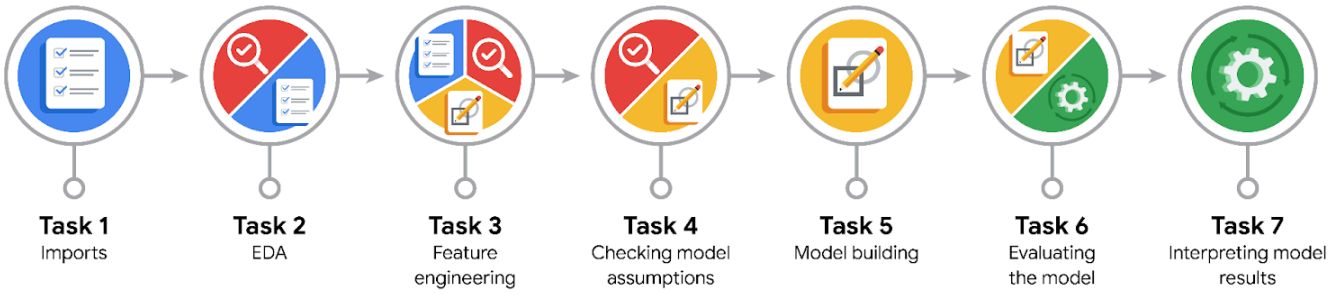
Relevant Interview Questions

Completing the end-of-course project will empower you to respond to the following interview topics:

- What kinds of business problems would be best addressed by supervised learning models?
- What requirements are needed to create effective supervised learning models?
- What does machine learning mean to you?
- How would you explain what machine learning algorithms do to a teammate who is new to the concept?
- How does gradient boosting work?

Reference Guide:

This project has seven tasks; the visual below identifies how the stages of PACE are incorporated across those tasks.



Data Project Questions & Considerations

PACE: Plan Stage

- What are you trying to solve or accomplish?

The goal is to streamline the moderation process for user-reported TikTok content by developing a machine learning model that classifies videos as either claims (1) or opinions (0). Initially, a logistic regression model was considered, but in this phase, more advanced models were explored.

Two models—Random Forest and XGBoost—were tested, with Random Forest emerging as the champion model due to its superior performance on both validation and test data. This model will be used to prioritize content moderation by identifying claims that may require further review.

- Who are your external stakeholders that you will be presenting for this project?

The key external stakeholders for this project include:

- Mary Joanna Rodgers – Project Management Officer

- Maika Abadi – Operations Manager

- What resources do you find yourself using as you complete this stage?

Key Python libraries, functions, and packages were used. The resources I used on my local system include VS Code with Jupyter Notebook.

- pandas for data manipulation

- Additionally, I used GitHub for documentation and tracking.

- Do you have any ethical considerations at this stage?

The ethical considerations involve ensuring the model balances false negatives and false positives while prioritizing false negatives.

- Classifying a claim as an opinion (false negative) could remove it from further review, potentially allowing misinformation to spread.

- Using F1 score helps balance precision and recall, but recall should be emphasized to reduce the risk of missing harmful claims.

- The model should be regularly evaluated for biases to ensure fair and accurate classification across different types of content.

- Is my data reliable?

Yes, the data is reliable as it is complete, well-structured, and clean.

- It includes all necessary fields to support the classification task, such as claim_status, video_id, video_duration_sec, video_transcription_text, verified_status, and author_ban_status.

- Engagement metrics like video_view_count, video_like_count, video_share_count, video_download_count, and video_comment_count provide additional context for classification.

- Data preprocessing steps ensured consistency, handling missing values and duplicates appropriately.

- What data do I need/would like to see in a perfect world to answer this question?

The current fields are sufficient to answer the question, but additional data could enhance model performance and interpretability.

- Additional engagement metrics: Deriving new features such as interaction ratios (likes-to-views, shares-to-comments) or temporal engagement patterns (how quickly a video gains views or likes) could provide deeper insights.

- User behavior and content moderation history:

  - Number of times the author received a warning – Frequent warnings may indicate a higher likelihood of violating platform policies.

  - History of flagged content – Tracking how often an author's content was flagged as inappropriate or as a claim could provide useful patterns.

  - Report frequency – The number of times a video was reported and the total reports across all videos by the same author could indicate potential violations.

- Engagement authenticity – Identifying whether engagement is organic or from bots (e.g., repetitive comments, sudden spikes in views from low-quality accounts) could help distinguish misleading claims from genuine discussions.

While the current model performs exceptionally well, these features could improve its robustness and adaptability to evolving content patterns.

- What data do I have/can I get?

I have the tiktok.csv, which contains all the necessary and sufficient fields for analysis, model development, and evaluation.

- The dataset includes key features such as claim_status, video_id, video_duration_sec, video_transcription_text, verified_status, and author_ban_status.

- Engagement-related fields like video_view_count, video_like_count, video_share_count, video_download_count, and video_comment_count provide valuable insights.

- The dataset is structured to support both exploratory data analysis and machine learning model training.

- What metric should I use to evaluate success of my business/organizational objective? Why?

Recall is the primary evaluation metric as the organization prioritizes minimizing false negatives (Type II errors).

- A high recall ensures that fewer actual claims are misclassified as opinions.

- This is crucial because missing a true claim could result in failing to flag potentially harmful or misleading content.

- Prioritizing recall in GridSearchCV helps optimize the model to correctly identify claims for moderation.

- While precision is also important, the risk of letting misinformation go unchecked makes recall the key focus.

PACE: Analyze Stage

- Revisit “What am I trying to solve?”Does it still work? Does the plan need revising?

I am trying to solve the issue of backlogs on user reports regarding author content.

- In this phase, the focus is on cleaning, preprocessing, and analyzing the data to extract key insights before model construction.

- The plan still works as expected, progressing effectively through different phases.

- No revisions are needed at this stage, as we are on the right track.

- Does the data break the assumptions of the model? Is that ok, or unacceptable?

The data does not break the assumptions of the model, mainly because Random Forest and XGBoost do not have strict assumptions like linearity or normality. However, key considerations include:

- Large Data Requirement

  - Random Forest and XGBoost perform well with large datasets but may overfit on small datasets if not tuned properly.

  - The dataset contains 19,382 observations, which is sufficiently large for these models.

  - Regularization in XGBoost helps prevent overfitting.

- Class Balance

  - There is no major class imbalance:

    - Claim: 50.35%

    - Opinion: 49.65%

  - This ensures that the model does not favor one class over the other.

- Categorical and Numerical Data Handling

  - Both models can handle categorical and numerical data.

  - XGBoost requires categorical variables to be encoded (e.g., one-hot encoding or label encoding).

- Handling Missing Values

  - Random Forest can handle missing values using surrogate splits.

  - XGBoost natively learns optimal splits for missing values.

Since these models do not assume linearity, normality, or homoscedasticity, the data's structure aligns well with their capabilities. Proper feature engineering and hyperparameter tuning ensure optimal performance.

- Why did you select the X variables you did?

The selected X variables were chosen based on their potential to influence whether a TikTok video is classified as a claim or an opinion, ensuring a mix of content-based, engagement-based, and metadata features.

- Content-Based Features

 - video_transcription_text: Tokenized to extract relevant linguistic patterns that may indicate a claim.

 - text_length: Longer texts may contain more detailed claims, while shorter ones may lean towards opinions.

- Engagement-Based Features

 - video_view_count: Claims might attract more views due to their controversial or attention-grabbing nature.

 - video_like_count: Higher likes could indicate perceived credibility or engagement.

 - video_share_count: Claims may be shared more as users spread information.

 - video_download_count: Downloading could suggest perceived importance or credibility.

 - video_comment_count: High comment activity may indicate a claim sparking debate.

- Metadata Features

 - video_duration_sec: Claims might be longer due to explanation or evidence presentation.

 - verified_status_verified: Verified users may be less likely to post false claims.

 - author_ban_status_banned / author_ban_status_under_review: Banned or under-review accounts might have a history of misinformation.

- Tokenized Text Features

 - Extracted claim-related phrases like "news claim," "social media," and "learned media" to detect patterns in claim-based language.

By including these variables, the model captures linguistic cues, engagement patterns, and user metadata, providing a robust framework for distinguishing claims from opinions.

- What are some purposes of EDA before constructing a model?

EDA helps with preprocessing, cleaning, and structuring the dataset to ensure high-quality input for training, validation, and testing. It provides a clear understanding of both predictor and outcome variables, making model training more robust.

Key purposes of EDA include:

- Checking Assumptions: Ensures data meets necessary modeling assumptions (e.g., linearity, normality, independence).

- Understanding Data Structure: Examines variable distributions, summary statistics, and overall patterns.

- Identifying Data Issues: Detects missing values, outliers, and inconsistencies to improve data quality.

- Selecting Features and Target Variable: Helps choose the most relevant predictors (X) for the target variable (Y).

- Preprocessing and Feature Engineering: Includes data transformations, cleaning, and selection to enhance model performance.

By addressing these factors early, EDA improves model reliability and interpretability.

- What has the EDA told you?

Key Insights from EDA

- Class Distribution: The dataset is nearly balanced, with 50.35% labeled as "claim" and 49.65% as "opinion," meaning no major class imbalance issues exist.

- Text Length Differences: Claims have longer transcriptions on average (95.38 characters) than opinions (82.72 characters), suggesting text length could be a useful feature for classification.

- Missing Values: Few missing values were found, and since their impact was minimal, rows with missing values were dropped.

- Duplicates: The dataset was checked for duplicate records to ensure data integrity.

- Feature Distributions:

  - The text length distribution is approximately normal, with claims generally having higher text lengths than opinions.

  - Other numerical features like video views, likes, shares, and comments may correlate with claim classification and need further exploration.

- Data Suitability for Tree-Based Models: Since tree-based models (Random Forest, XGBoost) are robust to outliers, no imputation or transformations were needed based on outlier distribution.

Overall, EDA confirms that the dataset is clean, structured, and contains distinguishable patterns, making it well-suited for modeling. The observed differences in text length and engagement metrics (likes, views, shares) could provide strong predictive power for distinguishing between claims and opinions.

- What resources do you find yourself using as you complete this stage?

In this phase, the focus is on exploring and understanding the data, handling missing values, checking for duplicates, and visualizing distributions.

- Packages and Libraries Used:

 - Data Manipulation:

  - pandas: For loading, exploring, and manipulating data (e.g., data.head(), data.shape, data.describe(), data.isna(), data.dropna(), data.duplicated()).

  - numpy: For numerical operations and array manipulations.

 - Data Visualization:

  - matplotlib.pyplot: For creating basic visualizations (e.g., histograms, labels, titles).

  - seaborn: For enhanced visualizations (e.g., sns.histplot() for plotting distributions).

 - Data Preprocessing:

  - sklearn.feature_extraction.text.CountVectorizer: For tokenizing text data into n-grams (though primarily used in the Construct phase, it is initialized here).

Additionally, I used GitHub for documentation and tracking.

PACE: Construct Stage

- Do I notice anything odd? Is it a problem? Can it be fixed? If so, how?

I noticed that verified status isn't really an important feature, contrary to initial expectations in the previous phase. It was initially considered for inclusion as an interaction term or covariate.

- The feature importance chart shows that verified status has little to no impact on the model's performance.

- This is not a problem, as the model still performs well with other independent variables.

- No fix is needed since strong features like video engagement metrics contribute significantly to classification accuracy.

- Which independent variables did you choose for the model, and why?

I selected these independent variables to ensure a comprehensive approach to distinguishing TikTok videos as claims or opinions, incorporating content-based, engagement-based, and metadata-driven features.

- Content-Based Features

- video_transcription_text (Tokenized) ⟶ Extracted linguistic patterns that may indicate a claim.

- text_length ⟶ Longer texts might contain detailed claims, while shorter ones may lean toward opinions.

- Engagement-Based Features

- video_view_count ⟶ Claims might attract more views due to their attention-grabbing nature.

- video_like_count ⟶ Higher likes could indicate perceived credibility or engagement.

- video_share_count ⟶ Claims may be shared more frequently as users spread information.

- video_download_count ⟶ Downloading could suggest perceived importance or credibility.

- video_comment_count ⟶ A high number of comments may indicate a claim sparking debate.

- Metadata Features

- video_duration_sec ⟶ Claims might be longer due to explanation or evidence presentation.

- verified_status_verified ⟶ Verified users may be less likely to post false claims.

- author_ban_status_banned / author_ban_status_under_review ⟶ Users with a history of misinformation may be more likely to post claims.

- Tokenized Text Features

- Extracted claim-related phrases such as "news claim", "social media", and "learned media" to capture patterns in claim-based language.

By incorporating these variables, the model leverages a mix of linguistic cues, engagement patterns, and user metadata to effectively classify videos as claims or opinions.

- How well does your model fit the data? What is my model's validation score?

The model fits quite well to the data.

- The Random Forest model fits the data exceptionally well:

- High Recall (0.9948) ⟶ Correctly identifies 99.48% of actual claims, minimizing false negatives.

- High Precision (0.9995) ⟶ When predicting a claim, it is correct 99.95% of the time, minimizing false positives.

- Confusion Matrix ⟶ Only 5 misclassified samples out of 3,817 in the validation set and 5 misclassified samples in the test set, showing an extremely low error rate.

- Generalization ⟶ The model performs consistently well on both validation and test sets, indicating no overfitting.

- Feature Utilization ⟶ Leverages relevant features like views, likes, shares, and downloads, confirming it learns from meaningful data patterns.

- The validation score in GridSearchCV is the average score of the chosen evaluation metric across all cross-validation folds.

- Best Recall Score (Validation): 0.9948 ⟶ The model correctly identifies 99.48% of actual claims in the validation set.

- Precision (Validation): 0.9995 ⟶ When the model predicts a claim, it is correct 99.95% of the time.

- F1 Score ⟶ The F1 score is also very high, as it balances precision and recall.

- Can you improve it? Is there anything you would change about the model?

The model performs quite well based on the evaluation metrics.

- The high recall (0.9948) and precision (0.9995) indicate that the model effectively minimizes both false negatives and false positives.

- The confusion matrix shows only 10 misclassified samples across validation and test sets, suggesting a very low error rate.

- The model generalizes well across different datasets, demonstrating its robustness.

There is no immediate need for improvement. However, potential areas for further evaluation include:

- Testing on additional datasets to assess generalization further.

- Exploring new feature engineering techniques to extract additional meaningful features.

- Considering alternative model architectures, but only if future performance issues arise.

At this stage, no significant changes are required, but ongoing monitoring on new data can help identify opportunities for refinement.

- What resources do you find yourself using as you complete this stage?

Packages and Libraries Used

Data Manipulation:

- pandas: For feature engineering (e.g., creating text_length column, concatenating dataframes).

- numpy: For numerical operations and array manipulations.

- pickle: For saving and loading trained models (e.g., CountVectorizer, RandomForestClassifier, XGBClassifier).

Data Preprocessing:

- sklearn.feature_extraction.text.CountVectorizer: For tokenizing text data into n-grams and transforming text features into numerical representations.

Data Modeling:

Model Selection & Evaluation:

- sklearn.model_selection.train_test_split: For splitting data into training, validation, and test sets.

- sklearn.model_selection.GridSearchCV: For hyperparameter tuning and cross-validation.

- sklearn.metrics: For evaluating model performance.

  - classification_report: For generating precision, recall, F1-score, and accuracy metrics.

  - confusion_matrix: For creating confusion matrices.

  - ConfusionMatrixDisplay: For visualizing confusion matrices.

Models:

- RandomForestClassifier: For building the random forest model.

- XGBClassifier: For building the XGBoost model.

Feature Importance:

- xgboost.plot_importance: For visualizing feature importances in the XGBoost model.

Data Visualization:

- matplotlib.pyplot: For visualizing confusion matrices and feature importances.

- seaborn: For additional visualizations (if needed).

Additional Resources:

- GitHub: For documentation and tracking.

PACE: Execute Stage

- What key insights emerged from your model(s)? Can you explain the model?

Key Insights for Random Forest Model

- High Performance on Recall and Precision:
  - Recall: 0.9948 (correctly identifies 99.48% of actual claims).
  - Precision: 0.9995 (99.95% accuracy when predicting a claim).
- Low Misclassification Rates:
  - Only 5 false positives and 5 false negatives on the validation set.
- Feature Importance:
  - The most predictive features are user engagement metrics:
    - Views
    - Likes
    - Shares
    - Downloads
  - These features help effectively distinguish between claims and opinions.
- Generalization to Unseen Data:
  - The model performs consistently well on validation and test sets.

Explanation of the Random Forest Model

The model is a random forest classifier, an ensemble learning technique that combines multiple decision trees. Each tree is a weak learner that makes predictions using different subsets of features and training data.

- Decision Trees & Splitting:
  - Trees split based on decreasing Gini impurity, ensuring nodes become more homogeneous.
  - The final prediction is determined by a majority vote across all trees.
- Feature Importance:
  - Engagement metrics (views, likes, shares, downloads) were the most predictive.

- Text length and n-grams provided additional useful patterns.

- Hyperparameter Optimization:

- GridSearchCV was used to fine-tune parameters, ensuring high recall and precision while avoiding overfitting.

How the Random Forest Model Makes Predictions

1. Feature Engineering:

- Uses features such as:

- Engagement metrics (views, likes, shares, downloads).

- Text length of the video transcription.

- Categorical variables (e.g., verified status, author ban status) encoded as dummy variables.

- N-grams from the video transcription text (e.g., 2-grams and 3-grams).

2. Training:

- The model is trained on the training set using GridSearchCV to optimize hyperparameters.

3. Validation:

- Evaluated on the validation set to ensure generalization and accuracy.

- What are the criteria for model selection?

The criteria for model selection focus on minimizing false negatives (Type II errors) since the organization's priority is to ensure that claims are correctly identified for moderation.

Key Evaluation Metric:

- F1 Score: Balances both precision and recall, which is crucial for this task.

Impact of Misclassification:

- False Positives (Misclassifying opinions as claims):

- Could lead to unnecessary content moderation.

- Might reduce user trust by incorrectly flagging harmless opinions.

- False Negatives (Misclassifying claims as opinions):

  - More problematic since claims may contain misinformation, harmful content, or policy violations.

  - If such videos bypass moderation, they could spread widely, damaging user trust and platform credibility.

Ethical Considerations:

- Prioritizing Recall:

  - Ensures that as many claims as possible are flagged for further review.

  - Minimizes the risk of harmful content slipping through undetected.

- Trade-off Between False Positives and False Negatives:

  - While false positives lead to unnecessary moderation, false negatives could allow the spread of misinformation.

  - A high recall ensures fewer harmful claims are missed, making the system more effective in content moderation.

- Does my model make sense? Are my final results acceptable?

Yes, the model makes sense and is interpretable as it classifies observations into two distinct classes. The final results are acceptable since the model performs exceptionally well, achieving high recall and precision, minimizing false negatives, and generalizing effectively across validation and test sets.

- Do you think your model could be improved? Why or why not? How?

The model could be improved by incorporating new features, whether completely new or extracted from existing data. Further hyperparameter tuning and optimization could also enhance performance. Additionally, increasing the dataset size could help the model generalize better. While the model does occasionally misclassify some observations, its high accuracy ensures that the level of misclassification remains within acceptable organizational standards.

- Were there any features that were not important at all? What if you take them out?

Yes, several features have near-zero importance, meaning they contribute very little (or not at all) to the model's predictions. These include:

- willing_wager

- social_media

- read_media

- point_view

- news_claim

- media_claim

- learned_media

- internet_forum

- friend_learned

- colleague_learned

What Happens If You Remove Them?

1. Reduced Model Complexity: Removing these unimportant features can make the model more interpretable and computationally efficient.

2. Minimal Impact on Performance: Since these features contribute little to impurity reduction, removing them is unlikely to significantly change accuracy, precision, or recall.

3. Better Generalization: With fewer, more relevant features, the model might generalize better to unseen data and avoid overfitting to noise.

- What business/organizational recommendations do you propose based on the models built?

I would recommend that even if the model is very accurate, human moderators or other review processes should still check content classified as a claim. This ensures that claims are properly verified before any further actions are taken, reducing the risk of false positives where an opinion is mistakenly classified as a claim. Since the company is willing to take the load for false positives, an additional verification step would help prevent any unintended consequences.

- Improve Model Performance by Removing Unimportant Features

Removing features with near-zero importance (e.g., social media, read media, point view) can streamline the model and improve efficiency.

- Implement Real-Time Claim Detection & Content Labeling

The model successfully distinguishes claims from opinions, so integrating it into TikTok's content moderation workflow could enhance real-time flagging and review.

- Prioritize Moderation for High-Engagement Content

Since key features influencing classification include video duration, view count, likes, shares, and downloads, prioritizing moderation for highly engaged content can help address misinformation more effectively.

- Given what you know about the data and the models you were using, what other questions could you address for the team?

What makes a video get more engagement?

Further NLP analysis could help identify specific text patterns in video transcripts that drive higher engagement.

- What patterns exist in user behavior related to claims?

Analyzing user interactions (likes, shares, comments) can reveal how users engage with claim-based content versus opinion-based content.

- What types of claims are most prevalent?

Categorizing claims by topic (e.g., politics, health, finance) could provide insights into the dominant themes in reported content.

- Can claim-based content be detected earlier?

Investigating early indicators, such as rapid engagement spikes or specific linguistic markers, could help in proactive moderation.

- What resources do you find yourself using as you complete this stage?

Execute Phase

Data Manipulation:

- pandas

- numpy

Data Modeling:

- sklearn.metrics (confusion_matrix, ConfusionMatrixDisplay, classification_report)

- RandomForestClassifier

- XGBClassifier

Data Visualization:

- matplotlib.pyplot

- seaborn (if needed)

Feature Importance:

- RandomForestClassifier.feature_importances_

I used Github for documentation and tracking.

- Is my model ethical?

It is. The data used is anonymized, and the classes were balanced, ensuring no bias. Removing claim-based content from the platform is beneficial. While more work can be done to improve accuracy in distinguishing opinions from claims, the model currently performs well.

- When my model makes a mistake, what is happening? How does that translate to my use case?

When the model makes a mistake, it misclassifies a user report as either a claim or an opinion. This results in two types of errors:

- False Negatives (Type II Error): A claim is misclassified as an opinion. This is the most critical issue because it means potentially harmful or misleading content goes unchecked, leading to misinformation risks.

- False Positives (Type I Error): An opinion is misclassified as a claim. This causes unnecessary human moderation, slowing down the process and frustrating content creators.

The organization prioritizes reducing false negatives to prevent misinformation from spreading. While the model shows no significant bias due to class balance, further analysis is needed to ensure no over-reliance on specific features.

For example:

- A false positive mislabels a safe, opinion-based video as a claim, causing unnecessary review and frustration for the creator.

- A false negative allows a claim (which could be misinformation or harmful content) to remain unchecked, risking the platform's reputation and user safety.