Roadmap Synthetic Data Generation Project

1. **Define Your Research Questions**
   * Example: “How closely does the synthetic data resemble the real data (in terms of statistical properties and ML performance) when generated by existing Python libraries (SDV, YData, etc.)?”
   * Decide on *evaluation metrics* that address two key aspects:
     1. **Data Similarity** (distribution matching, correlations, etc.)
     2. **Predictive Utility** (train/test an ML model on real vs. synthetic data)
2. **Choose Tools & Libraries**
   * **SDV** (Synthetic Data Vault)
   * **YData** (ydata-synthetic)
   * (Optional) Another library or technique if you want a broader comparison (e.g., Copulas, CTGAN, or a small custom GAN architecture).
   * Make sure to note **versions** and relevant dependencies to ensure reproducibility.
3. **Environment and Version Control**
   * Create a dedicated environment (e.g., a conda environment or venv) with pinned library versions.
   * Use Git or other version control for your code and notebook(s).

**2. Data Selection & Preprocessing**

1. **Select a Real-World Dataset**
   * Preferably one that’s medium-sized (few thousand to 100k rows) so you can run generation multiple times without excessive compute.
   * Domain examples: healthcare (HIPAA constraints), finance (sensitive PII), or any dataset with some potential privacy concerns.
2. **Clean and Explore the Data** (EDA)
   * Inspect **missing values**, **outliers**, **data types** (categorical, numerical).
   * Decide how to handle missing data (e.g., imputation or dropping incomplete rows).
   * Generate summary statistics:
     + **Mean, Median, Std** for numeric columns
     + **Value counts** for categorical columns
     + **Correlation matrix** or pairplot to see relationships.
   * Document these findings—this baseline EDA will help compare real vs. synthetic data later.
3. **Split Data (Optional)**
   * If you plan to do a membership inference test, keep a hold-out portion of the real data. Otherwise, you can train generative models on the entire dataset or a partial training split.

**3. Synthetic Data Generation**

1. **Set Up Each Library**
   * **SDV**:
     + Identify which model type you’ll use (e.g., **CTGAN**, **Copula models**, or **GaussianCopula**).
     + Provide your data to the SingleTableMetadata or relevant schema object if needed.
   * **YData**:
     + Install ydata-synthetic and choose a relevant method (GAN, CGAN, etc.).
   * Keep track of hyperparameters (e.g., epochs, batch size) in a config file or notebook cell for transparency.
2. **Train Generative Models**
   * **Train each model** on the real dataset (or training split).
   * **Watch for Overfitting**: Keep an eye on logs if provided by the library.
   * Keep training times and any resource usage notes for your final discussion.
3. **Generate Synthetic Data**
   * Use the trained model(s) to generate a synthetic dataset of the same size as the real one (or a chosen size).
   * Save these synthetic datasets (e.g., as CSV or Parquet). Name them clearly (e.g., synthetic\_sdv.csv, synthetic\_ydata.csv).

**4. Evaluation: Data Similarity**

A thorough evaluation will strengthen your project. Below are common metrics and methods:

1. **Descriptive Statistics**
   * Compare **mean, median, variance** of each numeric feature in real vs. synthetic data.
   * Compare **category distributions** in categorical features (e.g., bar plots or chi-square test).
2. **Statistical Similarity Tests**
   * **Kolmogorov–Smirnov (K-S) test** for continuous features.
   * **Chi-square test** for categorical features.
   * **Correlation Analysis**: Plot correlation matrices side by side (real vs. synthetic) or compute correlation difference metrics.
3. **Visual Comparisons**
   * **Histograms** or **boxplots**: Overlay real vs. synthetic data distributions.
   * **Pairwise scatter plots** for numeric columns: See if synthetic data preserves relationships.
4. **Dimensionality Reduction** (Optional, for Visualization)
   * Use **PCA** or **t-SNE** on real vs. synthetic data to see if points cluster similarly.
5. **Distribution Metrics** (if time permits)
   * **Frechet Inception Distance (FID)** is common in image domains, but rarely used for tabular data.
   * If dealing with continuous tabular data, you might adapt Earth Mover’s Distance (Wasserstein distance).

**5. Evaluation: Predictive Utility**

This section addresses how well the synthetic data can serve as a proxy for the real data in ML tasks.

1. **Select a Predictive Task**
   * **Classification** (if your dataset has a target label) or **Regression** (if there’s a numeric target).
   * Example: If your dataset is about credit approvals, classification could be “approve or not approve.”
2. **Training & Testing Scenarios**
   * **Scenario A**: Train the ML model on real data and test on a real test set (baseline performance).
   * **Scenario B**: Train the ML model on **synthetic** data and test on the **real test set**. This shows if synthetic data can stand in for real data in training.
   * (Optional) **Scenario C**: Blend real + synthetic data for training and see if it improves performance or addresses class imbalance, etc.
3. **ML Model Selection**
   * Start with something straightforward, e.g., **Random Forest** or **XGBoost**.
   * Keep the same hyperparameters across both real and synthetic-trained models to ensure fairness.
4. **Compare Performance Metrics**
   * **Accuracy, Precision, Recall, F1-score** (for classification).
   * **RMSE, MAE, R²** (for regression).
   * Summarize differences in a table or bar chart: e.g., “Model trained on real data got 85% accuracy vs. synthetic data’s 82%.”
5. **Interpret the Results**
   * Larger gaps in performance might indicate missing dependencies or distributions in synthetic data.
   * Smaller gaps suggest synthetic data is capturing relevant patterns.

**6. (Optional) Basic Privacy Assessment**

If your project scope allows, add a brief **privacy check**:

1. **Membership Inference Risk**
   * Some libraries or tutorials exist for membership inference. In basic form, you train a binary classifier to guess whether a sample was from the training set or not.
   * A high success rate indicates potential overfitting or privacy leakage.
2. **Discussion of Real vs. Synthetic Release**
   * Qualitative analysis: “Given these results, synthetic data might be safe to share externally, but we still see some risk under advanced attacks.”

This step is not always mandatory but can add a valuable dimension to your report if privacy is a core motivation.

**7. Documentation & Presentation**

1. **Literate Programming Notebook**
   * Use Jupyter or Google Colab.
   * **Document each step** in Markdown cells: data loading, cleaning, synthetic generation, evaluations, and final comparisons.
2. **Result Visualization**
   * Include plots of real vs. synthetic distributions, correlation heatmaps, ML performance tables, etc.
   * This makes your findings more intuitive and publishable.
3. **Discussion & Limitations**
   * Discuss **why** certain models performed better or worse.
   * Possible limitations: small sample size, complex data that simpler models can’t capture, computational constraints.
   * Future work: maybe explore advanced or custom GANs if you get more computational resources.
4. **Conclusions**
   * Summarize the key takeaways:
     + Are the synthetic datasets “good enough” for modeling?
     + What are the main trade-offs?

**8. Project Timeline Example**

Here’s a rough timeline you might follow over a few weeks:

1. **Week 1**:
   * Finalize dataset selection, complete EDA, set up environment, choose libraries.
2. **Week 2**:
   * Preprocess data, train synthetic models using SDV/YData.
   * Generate synthetic datasets.
3. **Week 3**:
   * Conduct data similarity analysis (statistical tests, distribution comparisons).
   * Start setting up ML pipelines for evaluation.
4. **Week 4**:
   * Train and evaluate ML models on real vs. synthetic data.
   * Optional: privacy checks.
5. **Week 5**:
   * Aggregate results, create visualizations, and write up discussion/conclusions.
   * Prepare final notebook or presentation.

**Additional Tips & Best Practices**

* **Keep an Eye on Compute**:
  + When using GAN-based methods, reduce batch size or epochs if training is too long.
  + Start with smaller subsets of your dataset to test workflow and code, then move to the full dataset.
* **Version Your Synthetic Data**:
  + Each time you tune a hyperparameter or switch a library, you’ll get a slightly different synthetic dataset. Keep them labeled so you can compare systematically.
* **Focus on Reproducibility**:
  + Clearly list library versions, seeds for randomness, and environment details (Python version, OS, etc.).
* **Add a Reflective Section**:
  + Discuss how synthetic data might help in real scenarios, any regulatory considerations, or potential biases introduced by your generation process.

**Final Thoughts**

Following this roadmap will allow you to systematically **compare synthetic data** to real data on both **statistical properties** and **practical machine learning performance**. You’ll end up with a well-documented notebook that demonstrates:

1. How to generate synthetic datasets using **existing Python libraries**.
2. How to **quantitatively** evaluate the similarity of synthetic vs. real data.
3. Whether models trained on synthetic data can perform comparably to those trained on real data.

These insights will make for a **solid, publication-ready** project, particularly if you include visual clarity, rigorous metrics, and transparent discussion of limitations.