Generative AI for Statistical Modeling – Cheat Sheet

1. Core Generative AI Models & Their Applications

Large Language Models (LLMs) (e.g., ChatGPT, Claude, Gemini)

- Code generation for feature engineering & preprocessing
- Data augmentation via textual insights
- Explanation of model outputs in natural language
- Automated report generation

Generative Adversarial Networks (GANs) (e.g., StyleGAN, CycleGAN)

- Synthetic data generation for imbalanced datasets
- Augmenting training sets with realistic rare event data
- Creating counterfactual examples for interpretability

Variational Autoencoders (VAEs) (e.g., β-VAE, VQ-VAE)

- Dimensionality reduction with generative capabilities
- Learning latent representations of complex datasets
- Anomaly detection via reconstruction loss

Diffusion Models (e.g., Stable Diffusion, TimeDiffusion)

- High-quality synthetic time-series data generation
- Data augmentation for small datasets
- Controlled synthetic data generation with structured constraints

Foundation Models for Time Series (e.g., TimeGPT, Chronos)

- Forecasting with minimal historical data
- Detecting anomalies and pattern shifts in time-series data
- Transfer learning for improved accuracy in time-series forecasting

2. When to Use Which GenAI Technique?

Challenge	Best GenAI Solution
Missing feature engineering	LLMs for automated feature extraction
Data scarcity or class imbalance	GANs, VAEs, Diffusion Models for synthetic data
Complex time-series forecasting	TimeGPT, Diffusion Models
Model interpretability & explanations	LLMs for natural language summaries, SHAP for feature attribution
Need for counterfactual analysis	GANs, VAEs for scenario simulation

3. Best Practices for Using GenAI in Statistical Modeling

- **Start with a baseline model:** Always compare GenAI-enhanced models to traditional approaches.
- **Validate synthetic data:** Ensure synthetic samples do not introduce bias or unrealistic patterns.
- **Combine multiple approaches:** Use LLMs for feature engineering, GANs for synthetic data, and TimeGPT for forecasting.
- **Focus on interpretability:** Ensure AI-driven models are explainable using SHAP, LIME, or LLM-generated insights.
- **Leverage transfer learning:** Fine-tune pre-trained GenAI models for better domain-specific accuracy.

4. Quick Reference: Key Python Libraries

Task	Python Library
LLM-powered automation	`openai`, `transformers`
Synthetic data generation	`torch`, `tensorflow`, `ctgan`, `synthetic-data-vault`
Feature engineering	`featuretools`, `tsfresh`, `sklearn`
Forecasting & time-series	`statsmodels`, `prophet`, `gluonts`, `nixtla/`
Interpretability & explainability	`shap`, `lime`, `interpret`, `captum`

5. Example: GenAI-Enhanced Fraud Detection Workflow

1. **Baseline model:** Train a traditional fraud detection classifier (e.g., XGBoost)
2. **Feature Engineering:** Use ChatGPT to extract high-impact transaction patterns
3. **Synthetic Data:** Generate synthetic fraudulent transactions using GANs
4. **Forecasting & Trend Analysis:** Use TimeGPT to detect emerging fraud patterns
5. **Interpretability:** Apply SHAP + LLM-generated explanations to enhance trust

Next Steps:

- Apply one GenAI technique in your own projects
- Experiment with synthetic data augmentation
- Implement SHAP or LLM explanations for your models
- Explore `TimeGPT` for forecasting tasks

Keep Learning, Keep Experimenting!