Breakdown of sklearn.datasets.make_regression

Works to create synthetic datasets to test regression algorithms, for educational purposes, benchmarking machine learning models, or simulating data scenarios.

1. **n_samples** (int, default=100):
   - Sample Size
   - Description: The number of samples to generate.
   - Example: If I want to simulate a study with 1000 individuals, set `n_samples=1000`.

2. **n_features** (int, default=100):
   - Number of Predictor Variables
   - Description: The number of features (independent variables) in the dataset.
   - Example: If I was simulating a situation with 5 different predictors (like age, income, etc.), I would set `n_features=5`.

3. **n_informative** (int, default=10):
   - Number of Informative Features
   - Description: The number of features actually used to build the linear model.
   - Example: If only 3 out of 5 features are relevant to my prediction, I would set `n_informative=3`.

4. **n_targets** (int, default=1):
   - Number of Response Variables
   - Description: The number of targets. If this is greater than 1, then it's a multi-output regression.
   - Example: For predicting a single outcome like house price, I would use `n_targets=1`. For predicting both house price and time on the market, I would use `n_targets=2`.

5. **bias** (float, default=0.0):
   - Intercept
   - Description: The bias term in the linear model.
   - Example: If I know my model should have an intercept of 10, I would set `bias=10`.

6. **effective_rank** (int or None, default=None):
   - Rank of the coefficient matrix
   - Description: If not None, the number of singular vectors required to explain the data. This is the correlation among the features.
   - Example: For a low-rank scenario with super correlated features, I would set `effective_rank` to a value lower than `n_features`

   1. Perform SVD on the initial feature matrix to decompose it into its singular values and vectors.

https://math.stackexchange.com/questions/2867075/what-is-ranks-do-in-singular-value-decomposition-if-rank-k-others-than-k-fir

Given a matrix X of size mxn, SVD decomposes X into three matrices:

$$X = U*Sigma*V^T$$

- U: An mxn orthogonal matrix, where the columns of U are known as the left singular vectors of X. These vectors are orthogonal to each other and to the space. The left singular vectors are essentially the eigenvectors of XX^T.
- Sigma: An mxn diagonal matrix (though not square if \(m \neq n\)), with non-negative real numbers on the diagonal. These are the singular values of X, sorted in descending order. The singular values are the square roots of the eigenvalues of X^TX or XX^T. The number of non-zero singular values is equal to the rank of matrix X.
- V^T: The transpose of an nxn orthogonal matrix, where the columns of V (the rows of V^T) are the right singular vectors of X. These vectors are the eigenvectors of X^TX.

2. Simulate effective_rank by reducing the number of significant singular values. This simulates a scenario where only a subset of the features carries the majority of the information, mimicking multicollinearity and reducing the matrix's rank.

7. **tail_strength** (float, default=0.5):
   - Tail Strength of Singular Values
   3. Applying tail_strength: This involves further adjusting the non-zero singular values to simulate the desired decay. A linear decay can be simulated by linearly reducing the magnitude of each successive singular value from the largest to the kth value. The tail_strength parameter can influence the slope of this decay. For a simple linear decay, you could adjust each of the top k singular values by a factor that decreases linearly

8. **noise** (float, default=0.0):
   - Measurement Error / Noise
   - The standard deviation of the Gaussian (normal) noise applied to the output.
   - If my outcome measurements have a noise level with a standard deviation of 3, I would set `noise=3`.

9. **shuffle** (bool, default=True):
   - Randomization
   - Description: Shuffle the samples and the features.

10. **coef** (bool, default=False):
    - Returning the Coefficient

- Description: If True, the coefficients of the underlying linear model are returned.
- Example: To inspect the coefficients that were used to generate the data, set `coef=True`.

11. **random_state** (int, RandomState instance or None, default=None):
    - The random seed
    - Example: For reproducible results, set `random_state` to a fixed number like `random_state=123`.