Sklearn Model "Cheat Sheet" for Pipelines

Here are the most common and useful `sklearn` models you can plug directly into the `('model', ...)` step of your `Pipeline` .

Your `model_pipeline` structure:

```
model_pipeline = Pipeline(
    steps=[
        ("preprocessing", preprocessor),
        ("model", ...)  # Plug any of the models below here!
    ]
)
```

1. Regression Models (Predicting a Number)

Use these when your target `y` is a number (like `final_score` or `price`).

A. Linear Models (The Fast Baselines)

These models fit a simple line or plane to the data. They are fast and easy to interpret.

1. Linear Regression

- **What it is:** The simplest baseline model.
- **Import:** `from sklearn.linear_model import LinearRegression`

- **Use in Pipeline:** `("model", LinearRegression())`

2. Ridge Regression

- **What it is:** A "smarter" `LinearRegression` that is more stable and less prone to overfitting. A great default choice.
- **Import:** `from sklearn.linear_model import Ridge`
- **Use in Pipeline:** `("model", Ridge())`

3. Lasso Regression

- **What it is:** A special `LinearRegression` that can perform *feature selection* by setting useless features to zero.
- **Import:** `from sklearn.linear_model import Lasso`
- **Use in Pipeline:** `("model", Lasso())`

B. Non-Linear Models (The Powerhouses)

These models can learn complex, flexible patterns.

4. K-Nearest Neighbors (KNN)

- **What it is:** A simple model that predicts by "polling" the `k` most similar data points from the training set. (Requires your `StandardScaler` to work well!)
- **Import:** `from sklearn.neighbors import KNeighborsRegressor`
- **Use in Pipeline:** `("model", KNeighborsRegressor(n_neighbors=5))`

5. Decision Tree

- **What it is:** A non-linear model that learns a series of "if-then" questions (like a flowchart).

- **Import:** `from sklearn.tree import DecisionTreeRegressor`
- **Use in Pipeline:** `("model", DecisionTreeRegressor(random_state=42))`

6. Random Forest

- **What it is:** The "powerhouse" model. It builds *hundreds* of Decision Trees and averages their predictions. Very accurate and robust.
- **Import:** `from sklearn.ensemble import RandomForestRegressor`
- **Use in Pipeline:** `("model", RandomForestRegressor(random_state=42))`

7. Gradient Boosting

- **What it is:** Often the "competition winner." It builds a series of weak trees, where each new tree learns from the mistakes of the previous one.
- **Import:** `from sklearn.ensemble import GradientBoostingRegressor`
- **Use in Pipeline:** `("model", GradientBoostingRegressor(random_state=42))`

8. Support Vector Machine (SVR)

- **What it is:** A powerful model that tries to fit a "tube" around the data points. (Also requires your `StandardScaler` to work well!)
- **Import:** `from sklearn.svm import SVR`
- **Use in Pipeline:** `("model", SVR())`

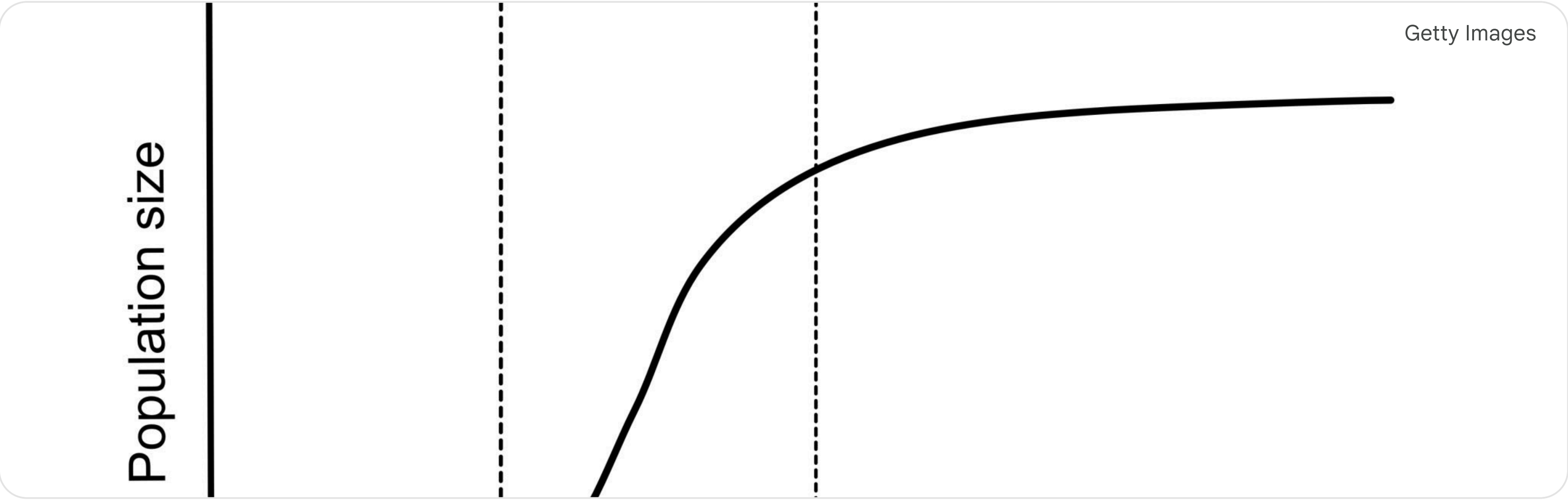
2. Classification Models (Predicting a Category)

Use these when your target `y` is a category (like `'Yes'/'No'` or `'Obesity_Type_I'`).

A. Linear Models (The Fast Baselines)

1. Logistic Regression

- **What it is:** The #1 baseline for classification. It predicts the *probability* of a class. It can handle binary ('Yes'/'No') and multi-class ('A'/'B'/'C') problems.



- **Import:** `from sklearn.linear_model import LogisticRegression`
- **Use in Pipeline:** `("model", LogisticRegression())`

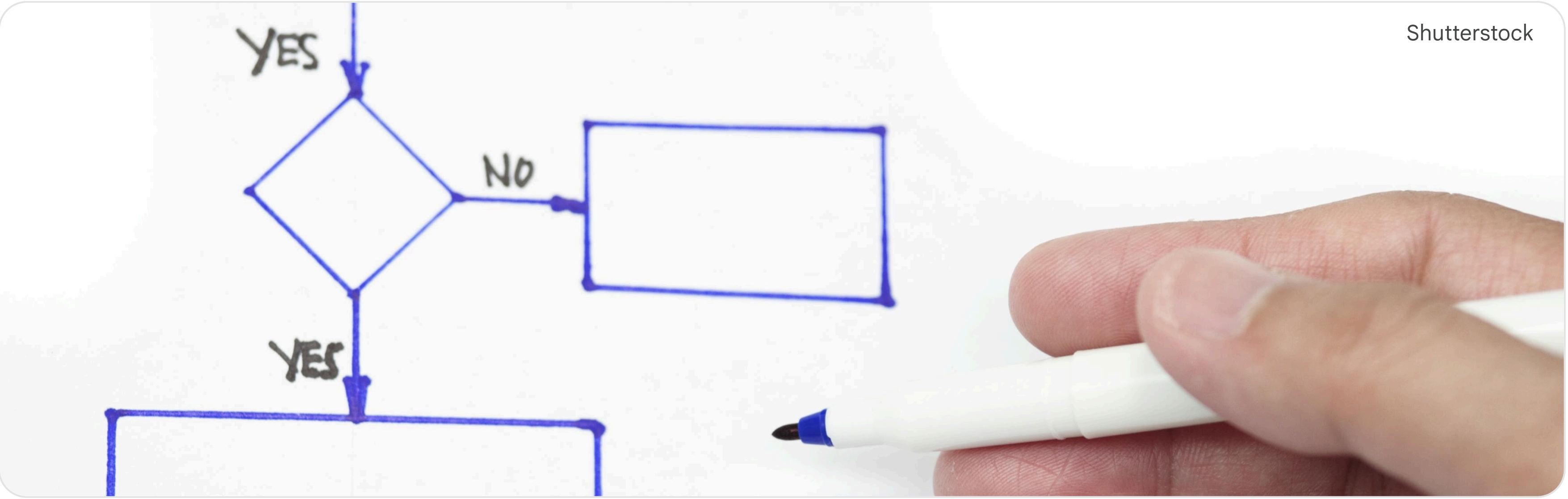
B. Non-Linear Models (The Powerhouses)

2. K-Nearest Neighbors (KNN)

- **What it is:** Predicts a category by taking a "majority vote" of the `k` most similar data points. (Requires `StandardScaler`!)
- **Import:** `from sklearn.neighbors import KNeighborsClassifier`
- **Use in Pipeline:** `("model", KNeighborsClassifier(n_neighbors=5))`

3. Decision Tree

- **What it is:** Learns a flowchart of "if-then" questions to decide on a class.



Shutterstock

- **Import:** `from sklearn.tree import DecisionTreeClassifier`
- **Use in Pipeline:** `("model", DecisionTreeClassifier(random_state=42))`

4. Random Forest

- **What it is:** The "powerhouse" classifier. Builds hundreds of trees and takes a majority vote. Excellent default choice.
- **Import:** `from sklearn.ensemble import RandomForestClassifier`

- **Use in Pipeline:** `("model", RandomForestClassifier(random_state=42))`

5. Gradient Boosting

- **What it is:** The "competition winner." Builds trees sequentially to fix each other's mistakes.
- **Import:** `from sklearn.ensemble import GradientBoostingClassifier`
- **Use in Pipeline:** `("model", GradientBoostingClassifier(random_state=42))`

6. Support Vector Machine (SVC)

- **What it is:** A very powerful model that finds the "widest possible margin" between classes. (Requires `StandardScaler`!)
- **Import:** `from sklearn.svm import SVC`
- **Use in Pipeline:** `("model", SVC(probability=True))` (Add `probability=True` if you need `predict_proba`)

7. Naive Bayes

- **What it is:** A simple, very fast probabilistic model that works surprisingly well, especially for text classification.
- **Import:** `from sklearn.naive_bayes import GaussianNB`