```
import numpy as np
from sklearn.linear_model import LinearRegression, Ridge
from sklearn.preprocessing import PolynomialFeatures
from sklearn.pipeline import make_pipeline
from sklearn.kernel_ridge import KernelRidge
from sklearn.metrics import mean_squared_error, r2_score

# 示例数据：替换为你的实际 A 和 B
# A: simulated spectra (100 x 5)
# B: real spectra (100 x 5)
np.random.seed(0)
A = np.random.rand(100, 5)
B = A @ np.array([[1.2, -0.5, 0.3, 0.7, -0.2]]).T + 0.1 * np.random.randn(100, 1)
B = np.hstack([B]*5)  # 模拟 5 个波段

def evaluate_model(name, model, A, B):
    B_pred = model.predict(A)
    mse = mean_squared_error(B, B_pred)
    r2 = r2_score(B, B_pred)
    print(f"{name} -> MSE: {mse:.6f}, R²: {r2:.4f}")
    return B_pred

# ------------------------------
# 1. 线性回归
linear_model = LinearRegression()
linear_model.fit(A, B)
evaluate_model("Linear Regression", linear_model, A, B)
```

```
# ------------------------------
# 2. Ridge 回归
ridge_model = Ridge(alpha=1.0)
ridge_model.fit(A, B)
evaluate_model("Ridge Regression", ridge_model, A, B)

# ------------------------------
# 3. 多项式回归（2 阶）
poly_model = make_pipeline(
   PolynomialFeatures(degree=2, include_bias=False),
   LinearRegression()
)
poly_model.fit(A, B)
evaluate_model("Polynomial Regression (deg=2)", poly_model, A, B)

# ------------------------------
# 4. Kernel Ridge Regression (Linear)
krr_linear = KernelRidge(kernel='linear', alpha=1.0)
krr_linear.fit(A, B)
evaluate_model("Kernel Ridge (linear)", krr_linear, A, B)

# 5. Kernel Ridge Regression (Polynomial)
krr_poly = KernelRidge(kernel='poly', degree=3, coef0=1, alpha=1.0)
krr_poly.fit(A, B)
evaluate_model("Kernel Ridge (poly deg=3)", krr_poly, A, B)

# 6. Kernel Ridge Regression (RBF)
krr_rbf = KernelRidge(kernel='rbf', gamma=0.5, alpha=0.01)
```

```
krr_rbf.fit(A, B)
evaluate_model("Kernel Ridge (RBF)", krr_rbf, A, B)

# 7. Kernel Ridge Regression (Sigmoid)
krr_sigmoid = KernelRidge(kernel='sigmoid', gamma=0.01, coef0=0.1, alpha=1.0)
krr_sigmoid.fit(A, B)
evaluate_model("Kernel Ridge (sigmoid)", krr_sigmoid, A, B)
```