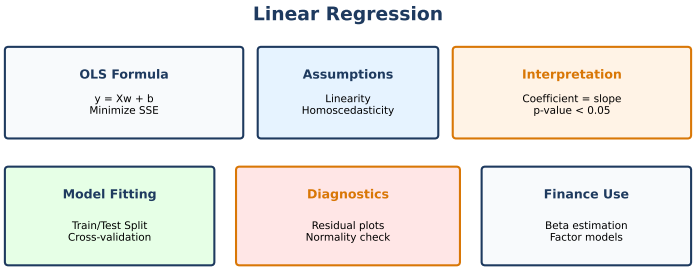
# Lesson 21 Summary: Linear Regression

## Data Science with Python – Key Concepts

Data Science Program

# Linear Regression Overview




*sklearn: LinearRegression().fit(X, y) | statsmodels: OLS for p-values*

**Linear regression is the foundation of predictive modeling**

# The OLS Method

**Ordinary Least Squares minimizes squared errors:**

- **Objective:** Minimize $\sum(y_i - \hat{y}_i)^2$
- **Solution:** Closed-form: $w = (X^T X)^{-1} X^T y$
- **Intercept:** Controls baseline prediction level

**Key insight:**
OLS finds the best linear combination of features to predict the target.

---

**OLS is the most common estimation method for linear regression**

# Model Assumptions

**Linear regression requires:**

- **Linearity:** True relationship is linear
- **Independence:** Observations are independent
- **Homoscedasticity:** Constant error variance
- **Normality:** Errors are normally distributed

---

**Violations of assumptions affect coefficient reliability**

# Coefficient Interpretation

**Understanding regression outputs:**

- **Coefficient:** Change in y per unit change in x
- **p-value:** Probability coefficient is zero
- **Confidence interval:** Range of plausible values

**Rule of thumb:**
p-value ¡ 0.05 typically considered significant.

**Use statsmodels for detailed statistical inference**

# Model Fitting in Python

**Two main approaches:**

- **sklearn:** Fast fitting, no p-values
- **statsmodels:** Full statistical output

**sklearn pattern:**

```
model = LinearRegression()
model.fit(X_train, y_train)
predictions = model.predict(X_test)
```

**Choose sklearn for ML, statsmodels for inference**

# Train-Test Split

**Prevent overfitting:**

- **Training set:** 70-80% of data for fitting
- **Test set:** 20-30% for evaluation
- **Random state:** Ensures reproducibility

**Never evaluate on training data!**

---

**Test set performance shows real-world capability**

# Residual Diagnostics

**Check model assumptions:**

- **Residual plot:** Should show no pattern
- **Q-Q plot:** Checks normality of errors
- **Scale-location:** Tests homoscedasticity

**Always visualize residuals before trusting results**

# Finance Application: Beta

**Estimating systematic risk:**

- **CAPM:** $r_i - r_f = \alpha + \beta(r_m - r_f) + \epsilon$
- **Beta:** Sensitivity to market movements
- **Alpha:** Excess return (manager skill)

**Beta estimation is a direct application of OLS**

# Common Pitfalls

**Avoid these mistakes:**

- **Multicollinearity:** Correlated features
- **Outliers:** Distort coefficient estimates
- **Overfitting:** Too many features

**Simple models often outperform complex ones**

# Quick Reference

**Essential Commands:**

| Task | Code |
|---|---|
| Split data | `train_test_split(X, y)` |
| Fit model | `LinearRegression().fit(X, y)` |
| Predict | `model.predict(X_new)` |
| Coefficients | `model.coef_, model.intercept_` |
| R-squared | `model.score(X, y)` |

**Master these basics before moving to regularization**