Lesson 25: Logistic Regression

Data Science with Python – BSc Course

45 Minutes

Learning Objectives

The Problem: Linear regression predicts continuous values, but what if we need to predict categories? How do we classify stocks as "buy" or "sell" based on features?

After this lesson, you will be able to:

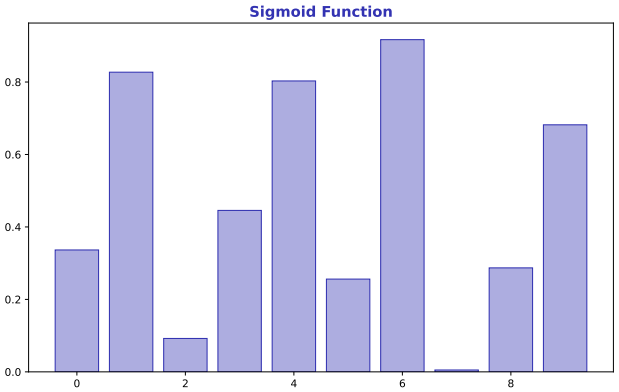
- Understand the sigmoid function and probability output
- Build binary classifiers with sklearn
- Interpret coefficients as odds ratios
- Predict market direction (up/down)

Finance Application: Predicting whether tomorrow's return is positive or negative

Sigmoid Function

From Linear to Probability

- $\sigma(z) = \frac{1}{1+e^{-z}}$ – squashes any value to (0, 1)
- Output is interpreted as $P(y = 1|x)$ – probability of positive class

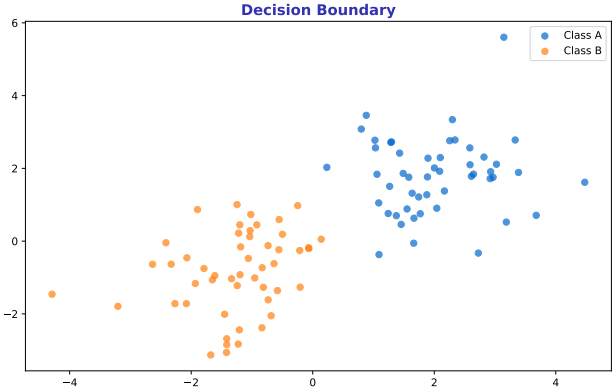


Key insight: Logistic regression = linear regression passed through sigmoid

Decision Boundary

Where Do We Draw the Line?

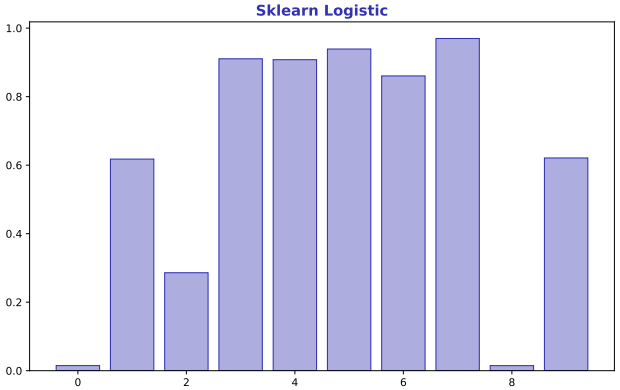
- Default threshold: predict class 1 if $P(y = 1) > 0.5$
- Decision boundary is where $\beta_0 + \beta_1 x = 0$



The boundary can be adjusted based on costs of false positives vs false negatives

Same API as Linear Regression

- `from sklearn.linear_model import LogisticRegression`
- `model.fit(X, y)` then `model.predict(X_new)`

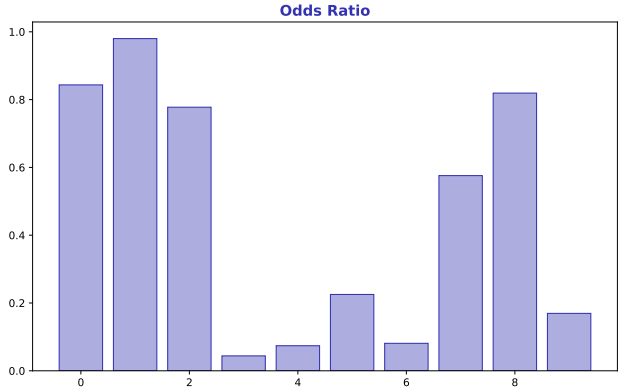


Use `model.predict_proba(X)` to get probabilities instead of 0/1

Odds Ratio Interpretation

What Do the Coefficients Mean?

- e^{β_j} = multiplicative change in odds per unit increase in x_j
- $e^{\beta} = 2$ means odds double when feature increases by 1

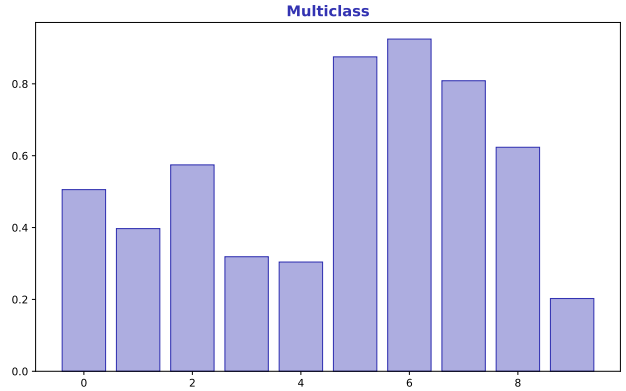


Finance: "Each 1% increase in momentum doubles the odds of positive return"

Multiclass Classification

More Than Two Classes

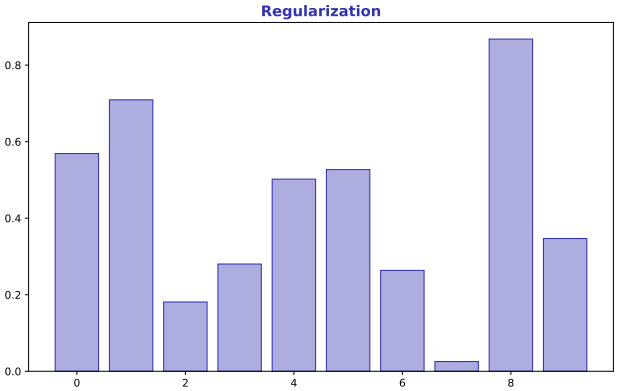
- One-vs-Rest (OvR): train K binary classifiers, pick highest probability
- Softmax/Multinomial: direct extension of sigmoid to K classes



sklearn default: `multi_class='auto'` **chooses automatically**

Preventing Overfitting

- Same L1 (Lasso) and L2 (Ridge) penalties apply
- sklearn default: L2 with `C=1.0` (inverse of λ)

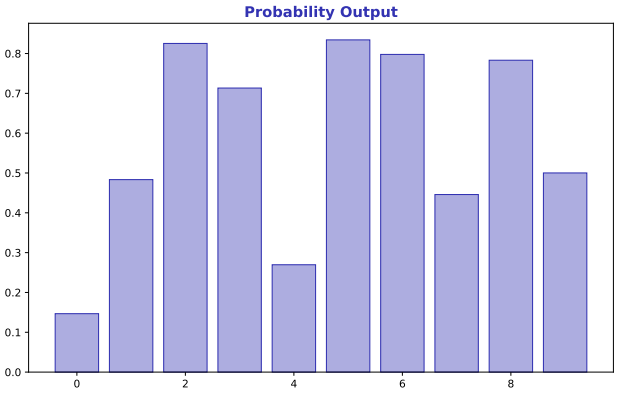


Lower C = stronger regularization (opposite of λ convention)

Probability Calibration

Are the Probabilities Accurate?

- Model says 70% – does event happen 70% of the time?
- Logistic regression is generally well-calibrated

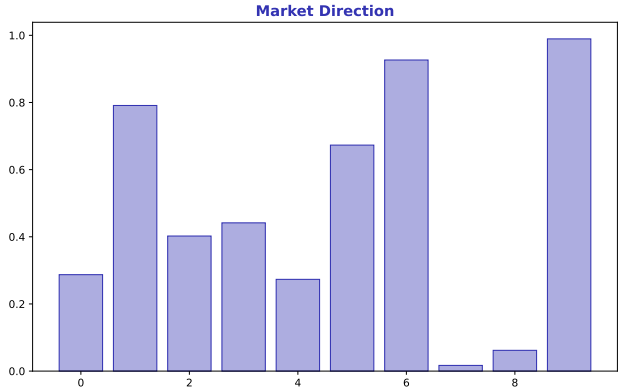


Calibration matters for risk management – you need accurate probabilities

Market Direction Prediction

Finance Application: Up or Down?

- Features: lagged returns, volume, volatility
- Target: 1 if next-day return > 0, else 0



Reality check: Even 52% accuracy can be profitable with proper sizing

Hands-On Exercise (25 min)

Task: Predict Stock Direction

1. Create binary target: 1 if next-day return > 0, else 0
2. Features: 5-day momentum, 20-day volatility, volume ratio
3. Fit logistic regression and examine coefficients
4. Calculate accuracy on held-out test set
5. Interpret: Which features increase odds of positive return?

Deliverable: Coefficient table with odds ratios + test accuracy.

Extension: Try different thresholds (0.4, 0.6) – how does accuracy change?

Lesson Summary

Problem Solved: We can now predict binary outcomes (up/down, buy/sell) and get probability estimates.

Key Takeaways:

- Sigmoid transforms linear output to probability
- Coefficients: e^{β} = odds ratio
- Same sklearn API: `fit()`, `predict()`, `predict_proba()`
- Regularization via C parameter (lower = stronger)

Next Lesson: Decision Trees (L26) – non-linear classification

Memory: Logistic = Linear + Sigmoid. Output is probability, not continuous value.