B.PHARM SEMESTER III · BP301T · UNIT II

BP301T Introduction to Machine Learning in Pharmaceutical Sciences

Unit II — Regression Models in Healthcare

Final Unit Notes

Continuous outcomes · Linear and multiple regression · Coefficients and residuals
MSE, RMSE and R^2 · Pharmaceutical applications · Python with sklearn

Pills & Python
Pharmacy • Data • AI • Healthcare

CHAPTER 2 — UNIT II: Regression Models in Healthcare

1. Learning Objectives

After studying this chapter, you will be able to:

- Explain what predictive modeling for continuous outcomes means.
- Describe simple linear regression and multiple linear regression in plain words.
- Interpret model coefficients (slope and intercept) with pharmacy examples.
- Explain residuals and goodness-of-fit.
- Calculate and interpret MSE, RMSE and R^2 score.
- Apply regression to pharmaceutical problems: dose–response, dissolution prediction, PK parameter estimation.
- Run a regression in Python using sklearn and explain every line and every output number.

2. Definitions

Regression. Regression is a supervised learning method that predicts a *continuous numerical* answer — any number on a scale, such as 62.5% dissolution, 4.2 hours half-life, or 120 mg dose requirement. "Continuous" means the answer can be any value, not just fixed categories.

Predictive modeling. Predictive modeling means building a mathematical model from past data so it can predict future values. A regression equation built from past dissolution tests that predicts tomorrow's batch result is a predictive model.

Linear regression. Linear regression draws the best straight line through the data points and uses that line for prediction. "Linear" = straight line; "regression" = predicting a number.

Multiple linear regression. Multiple linear regression is linear regression with *more than one* feature — it fits a flat plane (or hyperplane) instead of a single line. Example: predicting dissolution from *both* tablet hardness and coating thickness.

3. Terminology

Term	Plain meaning
Continuous outcome	An answer that is a measured number (dissolution %, blood drug level, half-life). Opposite of a category like yes/no.
Coefficient (slope)	How much the answer changes when one feature increases by 1 unit. The "rate" of change.
Intercept	The predicted answer when all features are zero — where the line crosses the Y-axis. The "starting point."
Residual	Actual value minus predicted value — the model's mistake on one record.
Goodness-of-fit	How closely the model's predictions match the real values overall.
MSE (Mean Squared Error)	Average of squared mistakes. Squaring punishes big mistakes heavily.
RMSE (Root Mean Squared Error)	Square root of MSE — the "typical mistake size" in the original units.
R^2 score	Fraction of the answer's variation explained by the model (0 to 1). 0.95 = the model explains 95%.
Dose–response relationship	How a drug's effect changes as the dose changes — a classic regression application.
PK parameters	Pharmacokinetic numbers (half-life, clearance, AUC) describing what the body does to the drug.

4. Explanation: predicting continuous outcomes

Technical concept. In regression, the label is a number. The model learns a formula, y = (coefficient × feature) + intercept, that best fits the training data, then plugs new feature values into the formula to predict.

Simple explanation. A pharmacist notices: "the harder the tablet, the slower it dissolves." Regression turns this observation into an exact formula: *Dissolution % = 90 − 0.5 × hardness*. Now for any new tablet's hardness, the pharmacist can calculate the expected dissolution without running the full test.

Example. Past data: tablets with hardness 5 kg dissolve 87.5% in 30 minutes; hardness 10 kg dissolve 85%. Regression finds the line connecting such points, then predicts dissolution for a new tablet of hardness 7 kg.

Diagram (draw this). Scatter plot: X-axis "tablet hardness (kg)", Y-axis "dissolution %". Five dots sloping downward. A straight line through the trend, labelled "regression line: the best straight-line summary." An arrow from a new X value (hardness = 7) up to the line and across to the Y-axis, labelled "prediction."

Key takeaway. **Regression = learning a formula from past numbers to predict future numbers.**

5. Linear regression and multiple linear regression

Technical concept — simple linear regression. With one feature x, the model is y = m·x + c. The algorithm chooses m (slope) and c (intercept) so the line sits as close as possible to all training points (least squares: minimising the sum of squared residuals).

Simple explanation. One input, one straight line. Example: only *dose* predicts *effect*.

Technical concept — multiple linear regression. With several features, the model is $y = m_1 \cdot x_1 + m_2 \cdot x_2 + \ldots + c$. Each feature gets its own coefficient, showing its individual contribution while the others are held constant.

Simple explanation. Several inputs, one flat surface. Example: dissolution predicted from *both* hardness and coating thickness — the model tells you the separate effect of each.

Table: simple vs multiple.

	Simple linear regression	Multiple linear regression
Number of features	One	Two or more
Shape	Straight line	Plane / hyperplane
Example	Effect from dose alone	Dissolution from hardness + coating thickness
sklearn tool	`LinearRegression`	`LinearRegression` (same tool, more columns)

Key takeaway. **One feature → line; many features → plane. The same sklearn tool handles both.**

6. Model coefficients and their interpretation

Technical concept. In $y = m \cdot x + c$: the intercept c is the predicted y when x = 0; the slope m is the change in predicted y for a one-unit increase in x. In multiple regression, each coefficient is the effect of that feature *keeping other features fixed*.

Simple explanation. The intercept is "where we start"; each slope is "how fast the answer moves when this input moves."

Worked pharmacy example. A fitted model for paracetamol tablet dissolution: *Dissolution % = 95 − 1.2 × hardness (kg).*

- Intercept 95: a (theoretical) zero-hardness tablet would dissolve 95%.
- Slope −1.2: each extra 1 kg of hardness *reduces* dissolution by 1.2 percentage points.
- Prediction for hardness 8 kg: 95 − 1.2 × 8 = 95 − 9.6 = **85.4%**.

Multiple regression example. *Dissolution % = 100 − 1.0 × hardness − 0.5 × coating (μm).*

- Holding coating fixed, 1 kg more hardness reduces dissolution by 1.0 point.
- Holding hardness fixed, 1 μm more coating reduces dissolution by 0.5 point.

Common mistake. Reading a coefficient as "proof" that changing the feature will *cause* the change. Regression shows association in the data, not proof of cause — a vital caution in pharmacy.

Key takeaway. **Intercept = starting point; slope = change in answer per one-unit change in the feature. Interpret the sign (+/−) and size, and remember association is not proof of cause.**

7. Residuals and goodness-of-fit

Technical concept. Residual = actual − predicted for one record. Goodness-of-fit describes how small the residuals are overall — small residuals mean the model fits well.

Simple explanation. The residual is "how far the model's guess missed the truth" for one case. If the real dissolution was 80% and the model said 82%, the residual is −2%.

Example. Our line predicted 60.5% for a 175 mg dose tablet, but the lab measured 62%. Residual = 62 − 60.5 = +1.5%. Small residuals across all tablets = good fit.

Diagram (draw this). Scatter plot with regression line; vertical dashed arrows from three dots to the line, labelled "residual = actual − predicted." Caption: "Short arrows = good fit; long arrows = poor fit."

Key takeaway. **Residuals are individual mistakes; goodness-of-fit is the overall report card.**

8. Performance metrics: MSE, RMSE, R^2

Technical concept.

- MSE = average of (residual^2). Units are squared, and big mistakes are punished heavily.
- RMSE = √MSE. Same units as the answer — "the typical prediction mistake."
- R^2 = 1 − (unexplained variation / total variation). Ranges 0–1 (can be negative for terrible models). R^2 = 0.95 means the model explains 95% of the variation in the answer.

Simple explanation. MSE/RMSE answer "how big are the mistakes on average?" (smaller = better). R^2 answers "how much of the ups and downs does the model explain?" (closer to 1 = better).

Worked example (verified by Python, hypothetical data). Dose (mg) vs dissolution % for 5 tablets: (50, 22), (100, 38), (150, 55), (200, 68), (250, 82). Fitted line: dissolution = 0.3 × dose + 8.

- MSE = **1.20** (average squared mistake)
- RMSE = **1.10%** (typical mistake about one percentage point — excellent)
- R^2 = **0.997** (the model explains 99.7% of the variation — an excellent fit)

Table: how to read the metrics.

Metric	Good value	Bad value	Tells you
MSE	Near 0	Large	Average squared mistake
RMSE	Near 0 (in answer's units)	Large	Typical mistake size
R^2	Near 1	Near 0 or negative	Fraction of variation explained

Common confusion. RMSE vs R^2: RMSE is in real units (mg, %, hours) — "how wrong in practice"; R^2 is unit-free — "how complete the explanation is." Report both.

Key takeaway. RMSE = typical mistake size (smaller is better); R^2 = share of variation explained (closer to 1 is better).

9. Regression in pharmacy: dose–response, dissolution, PK parameters

Dose–response relationships. How effect rises with dose is the oldest regression in pharmacy. A linear model on the suitable dose range estimates potency and predicts the effect of an untested dose. (Real dose–response curves are often S-shaped, so linear regression is used on the straight middle portion or after transformation — a point for viva.)

Predicting drug dissolution rates. From formulation and process features (hardness, disintegrant %, compression force), regression predicts % dissolved at 30 minutes — saving laboratory time and flagging risky batches early.

Estimating PK parameters. From patient features (age, weight, kidney function), regression can estimate clearance or half-life, supporting dose individualisation — e.g., predicting a longer half-life in elderly patients with reduced renal function.

Table: pharma regression applications.

Problem	Features	Label (number)	Benefit
Dose–response	Dose	% effect	Predict effect of new doses
Dissolution prediction	Hardness, coating	% dissolved at 30 min	Early batch quality warning
PK estimation	Age, weight, creatinine	Half-life (hours)	Individualised dosing
Stability	Temperature, humidity, time	% drug remaining	Shelf-life estimation

10. Python demonstration with sklearn (every number verified)

We use a small **hypothetical** dataset: dose (mg) vs % dissolution. Every output below was produced by actually running the code.

```
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

# Step 1: prepare the data (hypothetical example)
dose = np.array([50, 100, 150, 200, 250]).reshape(-1, 1)
dissolution = np.array([22, 38, 55, 68, 82])
```

Line-by-line (from zero): `import numpy as np` loads the number-handling library and nicknames it np. `from sklearn.linear_model import LinearRegression` brings in sklearn's regression tool. `np.array([...])` makes a list into a number table; `.reshape(-1, 1)` turns it into one column, because sklearn always wants features as a table with rows and columns. `dissolution` is our label — the answers.

```
# Step 2: build (train) the model
model = LinearRegression()
model.fit(dose, dissolution)
print("Slope:", model.coef_)
print("Intercept:", model.intercept_)
```

Output:

```
Slope: [0.3]
Intercept: 8.000000000000021
```

Explanation: `LinearRegression()` creates an empty model; `.fit(dose, dissolution)` trains it — the computer finds the best line. `model.coef_` is the slope: **0.3** — each extra 1 mg of dose raises dissolution by 0.3 percentage points. `model.intercept_` is **8.0** — the line's starting point. So the learned formula is: dissolution = 0.3 × dose + 8.

```
# Step 3: predict for a new dose
print(model.predict([[175]]))
```

Output: `[60.5]` — for a new 175 mg tablet the model predicts **60.5%** dissolution.

```
# Step 4: evaluate the model
predicted = model.predict(dose)
mse = mean_squared_error(dissolution, predicted)
print("MSE:", mse)
print("RMSE:", np.sqrt(mse))
print("R2:", r2_score(dissolution, predicted))
print("Residuals:", dissolution - predicted)
```

Output:

```
MSE: 1.2
RMSE: 1.0954451150103348
R2: 0.9973404255319149
Residuals: [-1.  0.  2.  0. -1.]
```

Explanation: `model.predict(dose)` gets the model's guesses for the training tablets. `mean_squared_error` compares guesses with truth → MSE **1.2**. RMSE ≈ **1.10%** — typical mistake about one percentage point. R^2 ≈ **0.997** — the line explains 99.7% of the variation: an excellent fit. Residuals (−1, 0, +2, 0, −1) are the individual misses — all tiny.

> **Multiple linear regression demo (hypothetical):** dissolution from dose *and* storage temperature:

```
X = np.array([[50,25],[100,25],[150,30],[200,30],[250,35]])  # [dose, temperature]
y = np.array([22, 38, 55, 68, 82])
multi = LinearRegression().fit(X, y)
print("Coefficients:", multi.coef_)
print("Intercept:", multi.intercept_)
print("R2:", multi.score(X, y))
```

Output:

```
Coefficients: [3.00000000e-01 1.02235471e-15]
Intercept: 8.0
R2: 0.9973404255319149
```

Explanation: dose coefficient **0.3** (same as before); temperature coefficient ≈ **0** (0.000… — essentially zero). Interpretation: in this hypothetical dataset, temperature adds no information beyond dose — the model honestly reports a near-zero effect. R^2 is again 0.997. *Lesson:* a near-zero coefficient means "this feature doesn't help here," not "the feature is useless in real life."

11. Key points

- Regression predicts continuous numbers; the label is a measured value.
- Simple linear regression: one feature, straight line, y = m·x + c.
- Multiple linear regression: several features, each with its own coefficient.
- Intercept = starting point; slope = change in answer per unit of the feature.
- Residual = actual − predicted; small residuals = good fit.
- RMSE = typical mistake size (smaller better); R^2 = variation explained (closer to 1 better).
- Pharmacy uses: dose–response, dissolution prediction, PK parameter estimation.
- sklearn: `LinearRegression().fit(X, y)` trains; `.predict()` predicts; metrics from `sklearn.metrics`.

12. Common confusions

- **Correlation vs causation:** a strong regression line shows association, not proof that the feature *causes* the outcome.
- **R^2 = 1 is suspicious:** a perfect 1.0 on real data often means overfitting or data leakage — investigate, don't celebrate blindly.
- **Intercept interpretation:** the intercept is the prediction at x = 0, which may be outside any realistic range (e.g., zero dose) — interpret with care.
- **Extrapolation:** predicting far outside the trained range (e.g., dose 2000 mg when trained on 50–250 mg) is unreliable.

13. Common mistakes

- Using regression for yes/no answers (that needs classification — Unit III).
- Forgetting `.reshape(-1, 1)` for a single feature in sklearn (it demands a 2-D table).
- Judging a model only by R^2 without checking RMSE in real units.
- Extrapolating beyond the data range.
- Treating a near-zero coefficient as "proof the feature never matters."

14. Mnemonic

"SIR" — **S**lope (rate of change), **I**ntercept (starting point), **R**esiduals (mistakes). **"R-R"** for metrics — **R**MSE = typical mistake (real units), **R^2** = share explained (0 to 1).

15. Infographic concept box — Unit II

- Title: "Regression in One Page"
- Left: scatter plot with best-fit line; arrow showing prediction for a new X.
- Center: the formula y = m·x + c with labels — m = "how fast", c = "where we start."
- Right: three gauges — RMSE (low = good), R^2 (near 1 = good), residuals (small = good).
- Bottom: three pharmacy icons — dose–response curve, dissolving tablet, blood-concentration curve.
- Footer: "Numbers in, numbers out — check the mistakes (RMSE), not just the fit (R^2)."

16. Exam summary

Regression is supervised learning for continuous (numerical) answers. Simple linear regression fits the best straight line y = m·x + c to one feature; multiple linear regression uses several features, each with its own coefficient. The intercept is the starting point and each slope tells how much the answer changes per unit of its feature. Residuals (actual − predicted) measure individual mistakes; goodness-of-fit is judged by MSE/RMSE (typical mistake size, smaller is better) and R^2 (fraction of variation explained, closer to 1 is better). In pharmacy, regression models dose–response relationships, predicts dissolution rates and estimates PK parameters such as half-life. In sklearn, `LinearRegression().fit(X, y)` trains the model, `.predict()` gives predictions, and `mean_squared_error` / `r2_score` evaluate it.

17. One-minute revision

- Regression → predicts numbers.
- y = m·x + c; m = slope, c = intercept.
- Multiple regression → many features, one coefficient each.
- Residual = actual − predicted.
- RMSE small = good; R^2 near 1 = good.
- Pharmacy: dose–response, dissolution, PK parameters.
- sklearn: fit → predict → evaluate.

18. Exam questions

Short answers:

1. Define regression. How is it different from classification?
2. Interpret: "Dissolution % = 95 − 1.2 × hardness." What does −1.2 mean?
3. Define residual. If actual = 80 and predicted = 83, what is the residual?
4. What do RMSE and R^2 tell you? Which units does RMSE carry?

Long answers:

5. Explain simple and multiple linear regression with pharmacy examples. How do you interpret each coefficient?
6. Describe MSE, RMSE and R^2 with formulas in words. Using the dose–dissolution example (RMSE ≈ 1.1%, R^2 ≈ 0.997), judge the model.
7. Explain three pharmaceutical applications of regression with features and labels for each.

MCQs:

8. Regression is used when the label is:
   a) yes/no b) a category c) a continuous number d) missing
   Answer: c
9. In y = 8 + 0.3x, if x increases by 10, y increases by:
   a) 0.3 b) 3 c) 8 d) 10.3
   Answer: b
10. R^2 = 0.95 means:
    a) 95% accuracy b) the model explains 95% of the variation c) 5% error rate d) 95 predictions correct
    Answer: b
11. In sklearn, the model is trained with:
    a) `model.predict(X)` b) `model.fit(X, y)` c) `model.score()` d) `LinearRegression.predict`
    Answer: b

Viva:

12. Your R^2 is 0.99 on training data but 0.55 on test data. What happened and what will you do?
13. Why can't we use linear regression to predict "ADR: yes/no"?

Application-based:

14. You have tablet hardness, disintegrant % and compression force, and want to predict % dissolved at 30 min for a new batch. Frame this as a regression problem: name features, label, and two metrics you would report.

19. References (Unit II)

1. James, G., Witten, D., Hastie, T. and Tibshirani, R. *An Introduction to Statistical Learning with Applications in Python*. Springer. Available at: statlearning.com.
2. Walters, S. J., Campbell, M. J. and Machin, D. *Medical Statistics: A Textbook for the Health Sciences*. Wiley-Blackwell.
3. Géron, A. *Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow*. O'Reilly Media.

COMBINED REFERENCES (all units)

As listed in the official BP301T syllabus (preferably latest editions):

1. James, G., Witten, D., Hastie, T. and Tibshirani, R. An Introduction to Statistical Learning with Applications in Python. Springer. Available at: statlearning.com.
2. Simon, G. J. and Aliferis, C. Artificial Intelligence and Machine Learning in Health Care and Medical Sciences: Best Practices and Pitfalls. Springer.
3. Brown, N. Artificial Intelligence in Drug Discovery. Royal Society of Chemistry.
4. Walters, S. J., Campbell, M. J. and Machin, D. Medical Statistics: A Textbook for the Health Sciences. Wiley-Blackwell.
5. Géron, A. Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow. O'Reilly Media.