PSYCH 201B

Statistical Intuitions for Social Scientists

Modeling data VI

You can download these slides:
course website > Week 7 > Overview

Today's Plan:
Part 1 (together)
Part 2 (notebooks on your own)

02/19/2025

Announcements

Announcements

1. All notebooks from this week 1-7 **due Friday**
2. Final Project

    Feb 24 - Proposal template provided

    Mar 12 - Proposal due (meet with us before this!)

    Mar 20 - Final due (propose early, finish early!)

3. HW 3 will be posted *tonight*

    due before lab Feb 25th (next Tues)

Today’s Plan

1. First Half (together)
   - Treatment coding review
   - Treatment with 3 levels
2. Second Half (on your own)
   - Notebooks 4, 5, **6, 7**
   - Look at previous notebook solutions if you haven’t

How does the GLM see **categorical variables**?

We encode levels of a categorical variable using numbers

How does the GLM see **categorical variables**?

We encode levels of a categorical variable using numbers

We represent ***k levels*** of a categorical variable with ***k-1 parameters*** using one of many possible **coding schemes**

Treatment (Dummy) Coding 2-levels

- **Reference level** is coded as 0, and other level is coded as 1
- **Intercept** = reference (mean); **Slope(s)** = mean difference from reference
- Default when using `C()` in statsmodels and `lm()` in R

Yes = 1
No = 0

Balance	Student
i64	str
16	"Yes"
1216	"Yes"
148	"No"
108	"No"
532	"Yes"

Data

```
array([[1., 1.],
       [1., 1.],
       [1., 0.],
       [1., 0.],
       [1., 1.]])
```

Design Matrix

Treatment (Dummy) Coding 2-levels

```python
# Treat "Student" as a categorical variable
a_model = ols('Balance ~ C(Student)', data=df.to_pandas())
```

Yes = 1
No = 0

Balance	Student
i64	str
16	"Yes"
1216	"Yes"
148	"No"
108	"No"
532	"Yes"

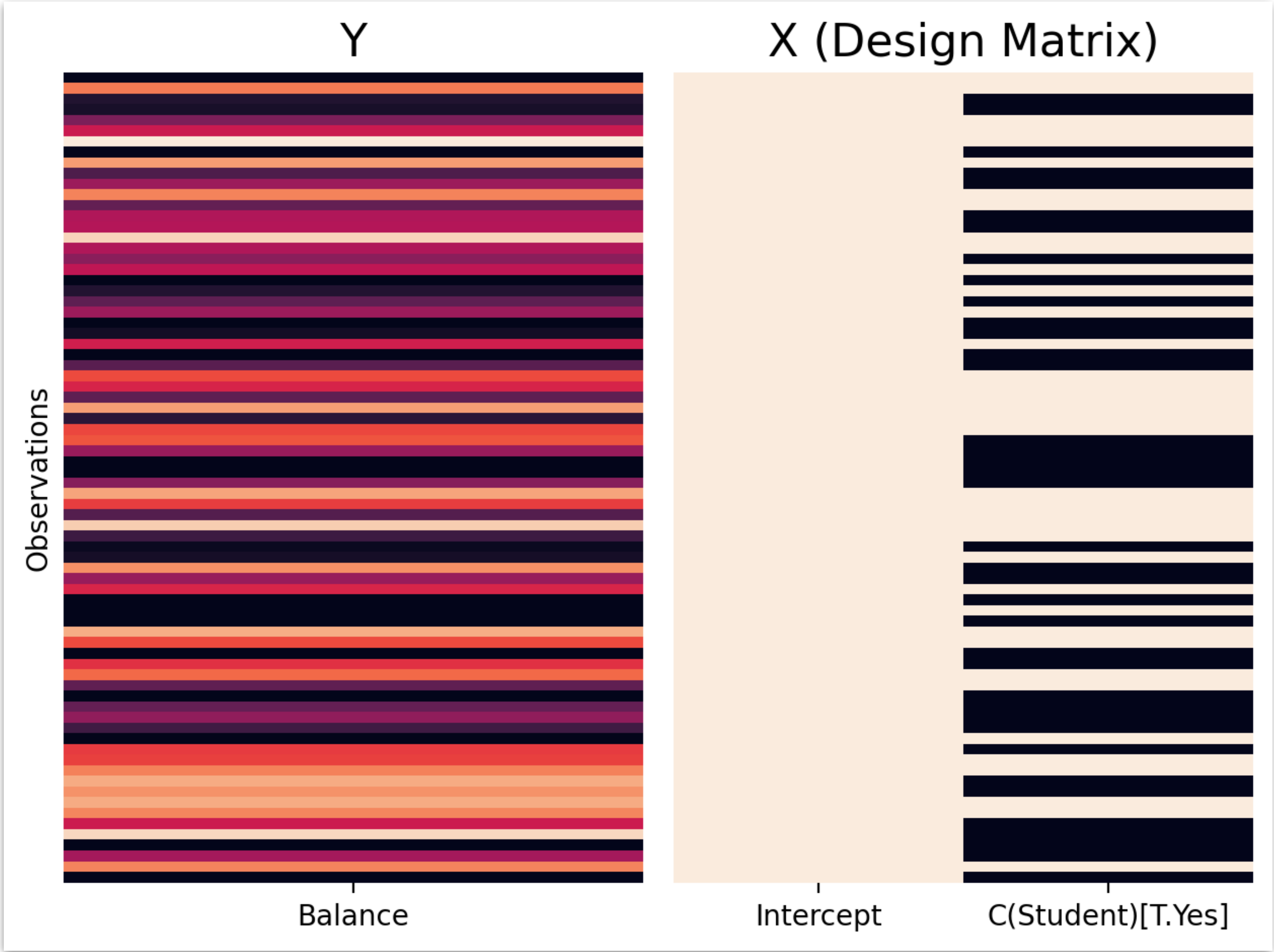
Data

```
array([[1., 1.],
       [1., 1.],
       [1., 0.],
       [1., 0.],
       [1., 1.]])
```

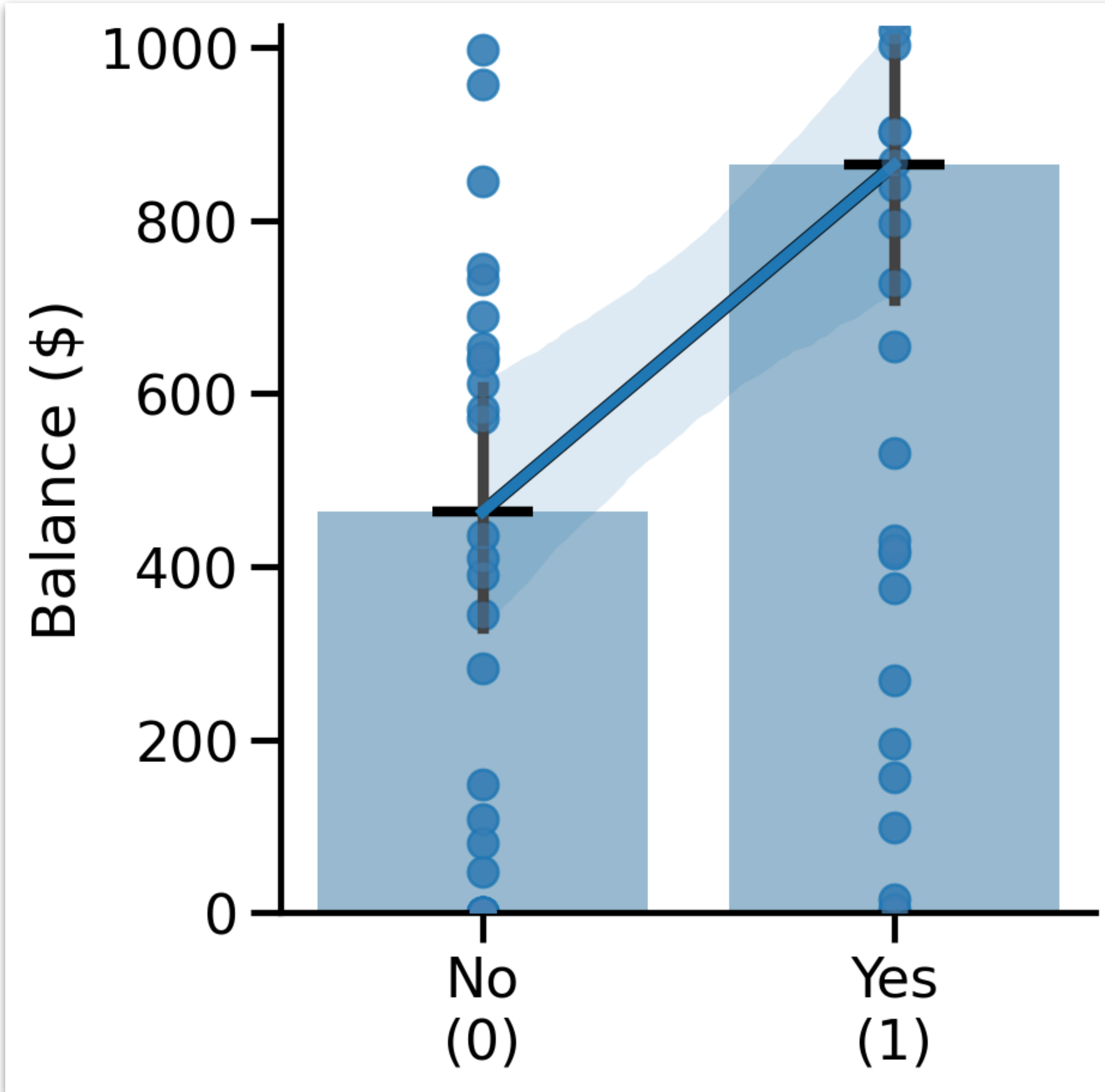
Design Matrix

Treatment (Dummy) Coding 2-levels

```python
# Treat "Student" as a categorical variable
a_model = ols('Balance ~ C(Student)', data=df.to_pandas())
```



Linear model of mean difference aka "independent t-test"



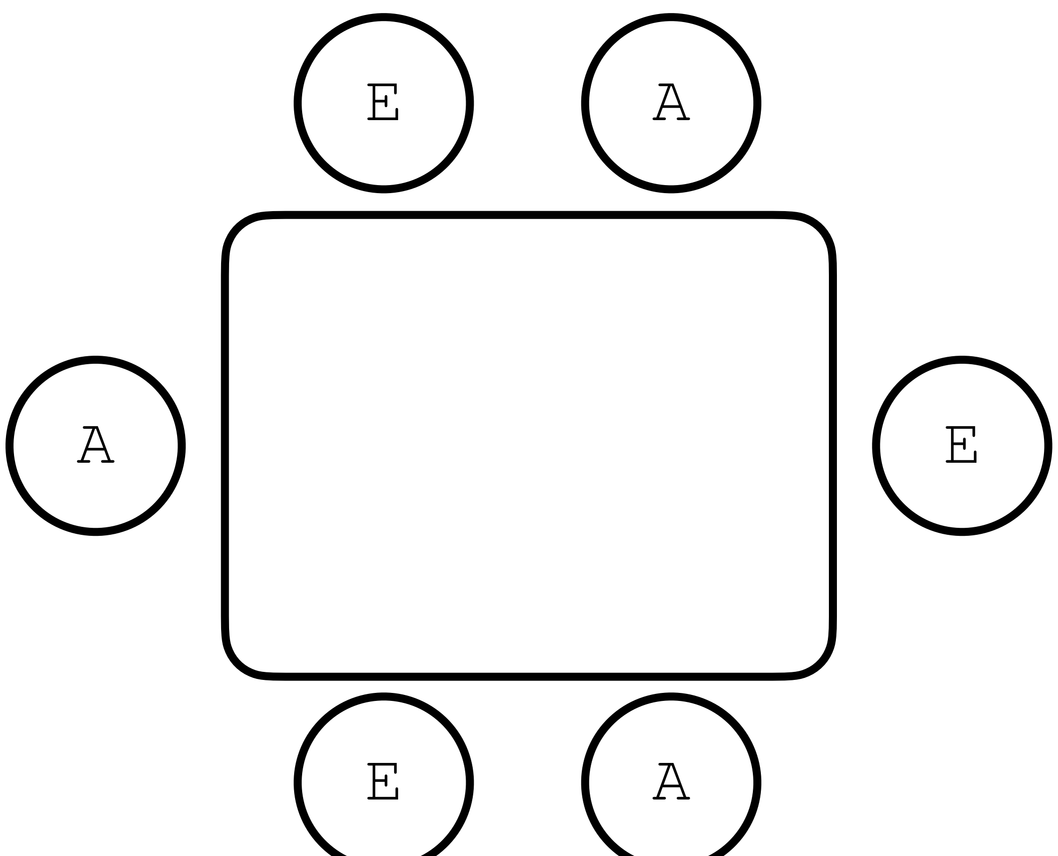
- **Reference level** is coded as 0, and other level is coded as 1
- **Intercept** = reference (mean); **Slope(s)** = mean difference from reference
- Default when using `C()` in statsmodels and `lm()` in R

Categorical predictors (3+ levels)

New Dataset

Abstract

Adopting a quasi-experimental approach, the present study examined the extent to which the influence of poker playing skill was more important than card distribution. Three average players and three experts sat down at a six-player table and played **60 computer-based** hands of the poker variant ‘‘Texas Hold’em’’ for money. In each hand, one of the average players and one expert received (a) better-than-average cards (winner’s box), (b) average cards (neutral box) and (c) worse-than-average cards (loser’s box). The standardized manipulation of the card distribution controlled the factor of chance to determine differences in performance between the average and expert groups. Overall, 150 individuals participated in a ‘‘fixed- limit’’ game variant, and 150 individuals participated in a ‘‘no-limit’’ game variant.



During the game, one expert player and one average player received

(a) the winning hand 15 times and the losing hand 5 times (winner’s box condition)

(b) the winning hand 10 times and the losing hand 10 times (neutral box condition)

(c) the winning hand 5 times and the losing hand 15 times (loser’s box condition)

Dataset

skill	hand	limit	balance
expert	bad	fixed	4.00
expert	bad	fixed	5.55
expert	bad	none	5.52
expert	bad	none	8.28
expert	neutral	fixed	11.74
expert	neutral	fixed	10.04
expert	neutral	none	21.55
expert	neutral	none	3.12
expert	good	fixed	10.86
expert	good	fixed	8.68

skill = expert/average

hand = bad/neutral/good

limit = fixed/none

balance = final balance in Euros

2 (skill) x 3 (hand) x 2 (limit) design

25 participants per condition

n = 300

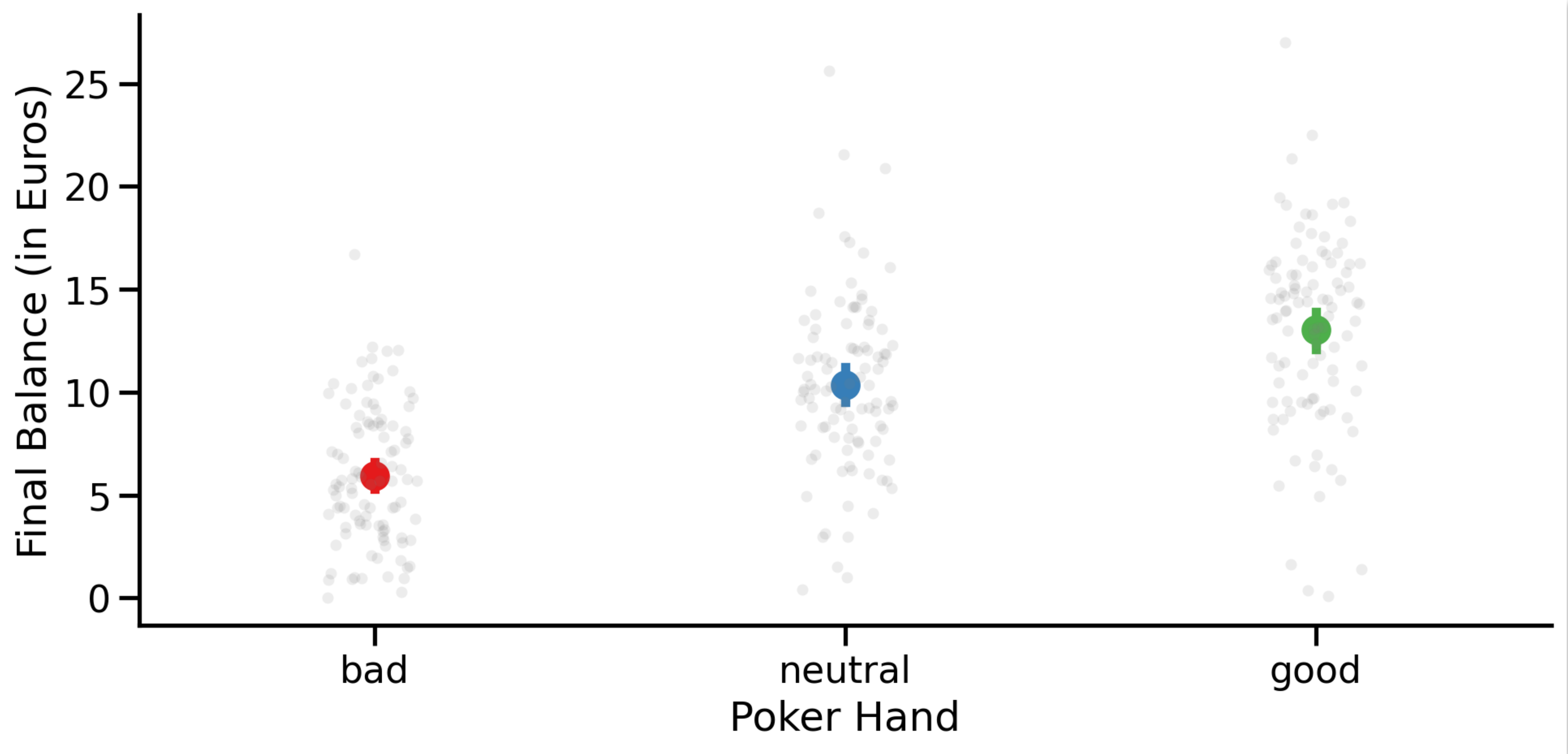
Meyer, G., von Meduna, M., Brosowski, T., & Hayer, T. (2012). Is poker a game of skill or chance? A quasi-experimental study. *Journal of Gambling Studies*

Do better hands win more money?

participant	skill	hand	limit	balance
1	expert	bad	fixed	4.00
2	expert	bad	fixed	5.55
26	expert	bad	none	5.52
27	expert	bad	none	8.28
51	expert	neutral	fixed	11.74
52	expert	neutral	fixed	10.04
76	expert	neutral	none	21.55
77	expert	neutral	none	3.12
101	expert	good	fixed	10.86
102	expert	good	fixed	8.68

hand = {bad, neutral, good}

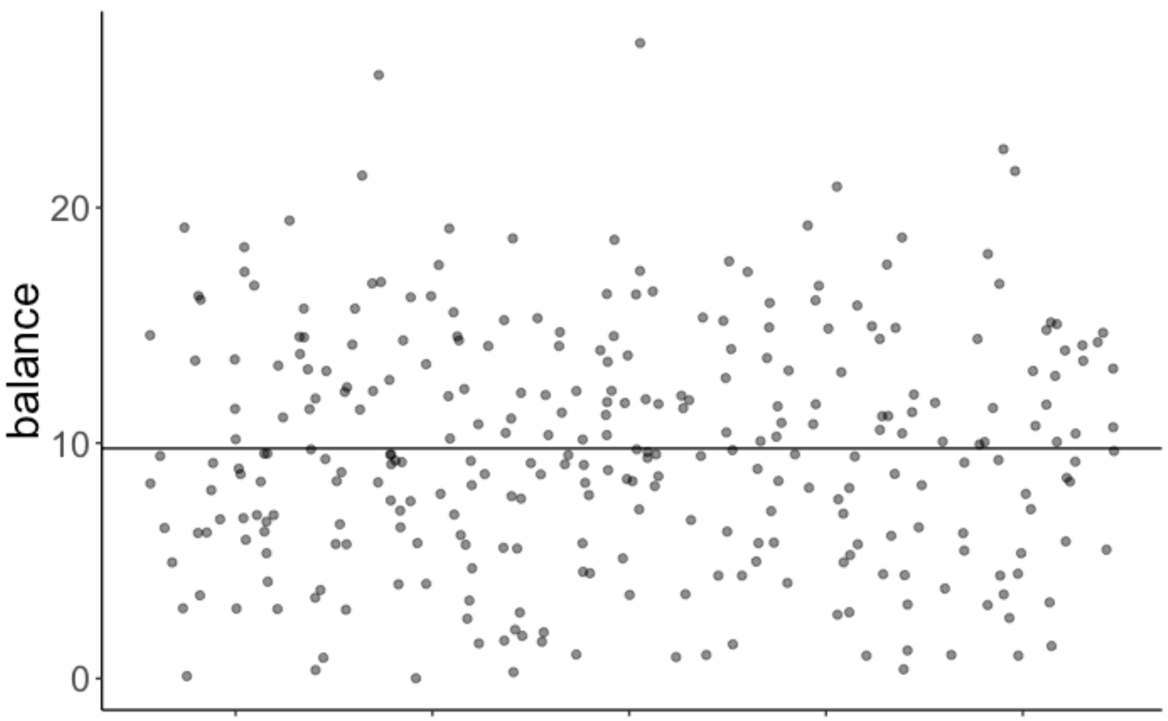
Visualize the data first



H_0: Card quality does not affect the final balance

Model C

$$Y_i = \beta_0 + \epsilon_i$$

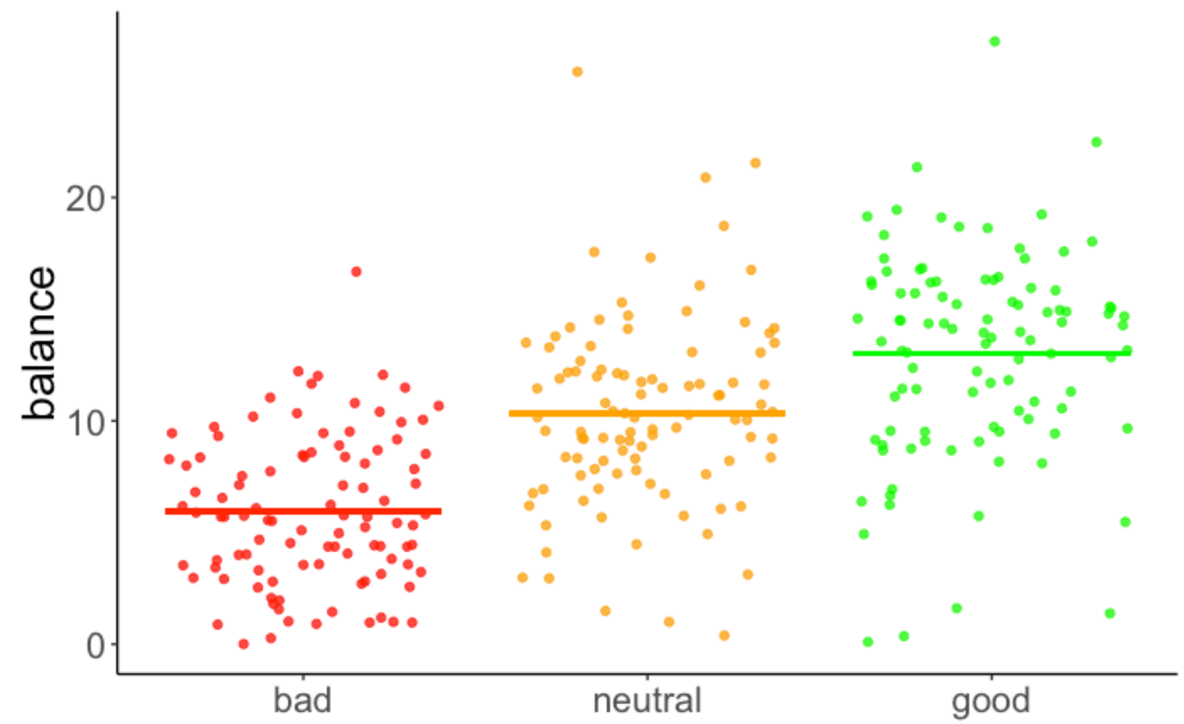


H_1: Students and non-students have different balances.

Model A

$$Y_i = \beta_0 + \beta_1 X_i + \epsilon_i$$

hand



Worth it?

```
# Compact
model_c = ols('balance ~ 1', data=df.to_pandas())
results_c = model_c.fit()

# Augmented
model_a = ols('balance ~ C(hand)', data=df.to_pandas())
results_a = model_a.fit()

# Worth it?
anova_lm(results_c, results_a)
```

Worth it!

	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)
0	299.0	7579.984625	0.0	NaN	NaN	NaN
1	297.0	5020.583223	2.0	2559.401402	75.702581	2.699281e-27

You just did a One-way ANOVA!

```
# Compact
model_c = ols('balance ~ 1', data=df.to_pandas())
results_c = model_c.fit()

# Augmented
model_a = ols('balance ~ C(hand)', data=df.to_pandas())
results_a = model_a.fit()

# Worth it?
anova_lm(results_c, results_a)
```

Worth it!

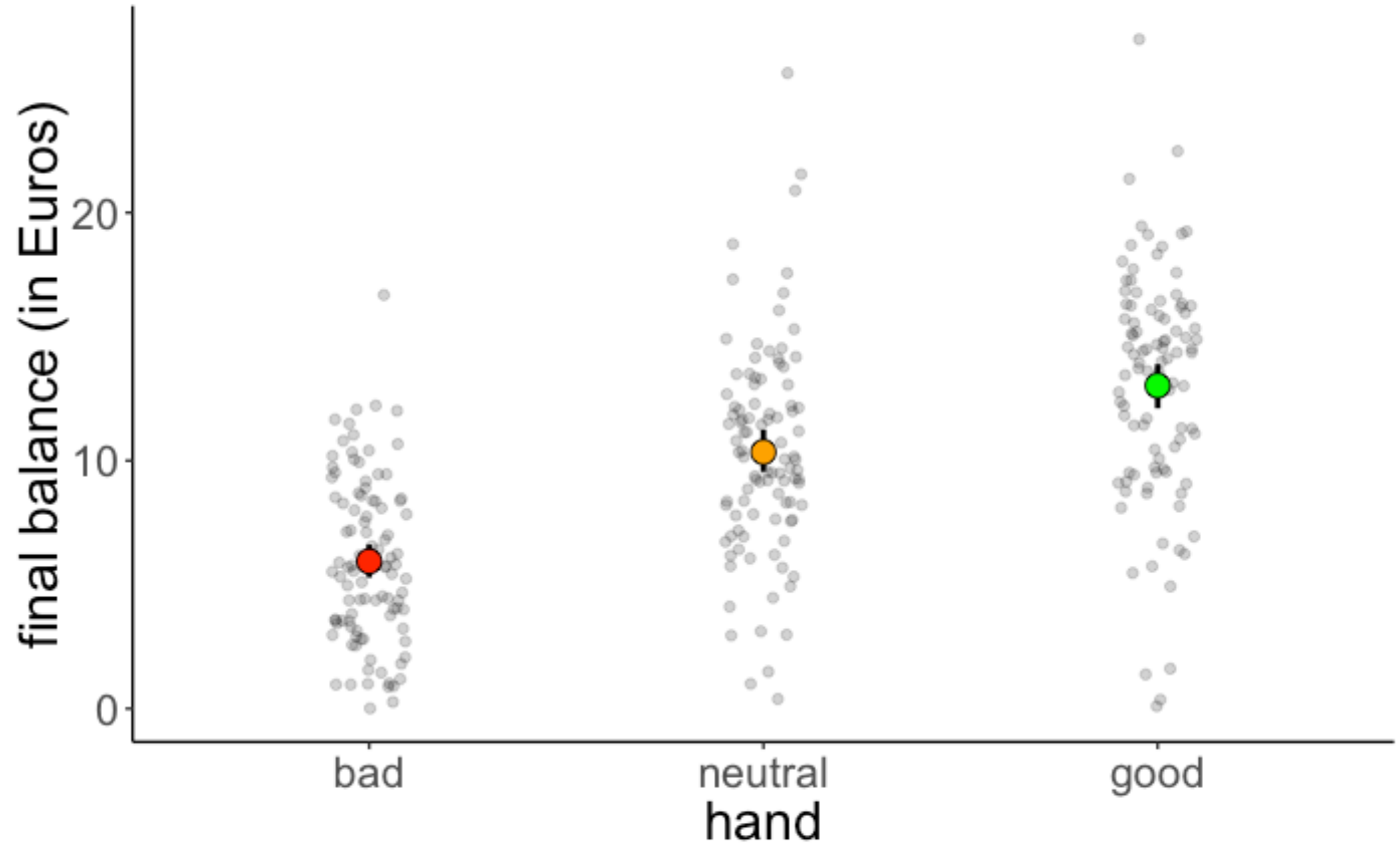
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)
0	299.0	7579.984625	0.0	NaN	NaN	NaN
1	297.0	5020.583223	2.0	2559.401402	75.702581	2.699281e-27

```
1  anova_lm(results_a, type=3)
```

✓ 0.0s

	df	sum_sq	mean_sq	F	PR(>F)
C(hand)	2.0	2559.401402	1279.700701	75.702581	2.699281e-27
Residual	297.0	5020.583223	16.904321	NaN	NaN

Reporting an ANOVA



The final balance differed significantly as a function of the quality of a player's hand (i.e. whether the hand was bad, neutral, or good), $F(2, 297) = 75.703$, $p < .001$.

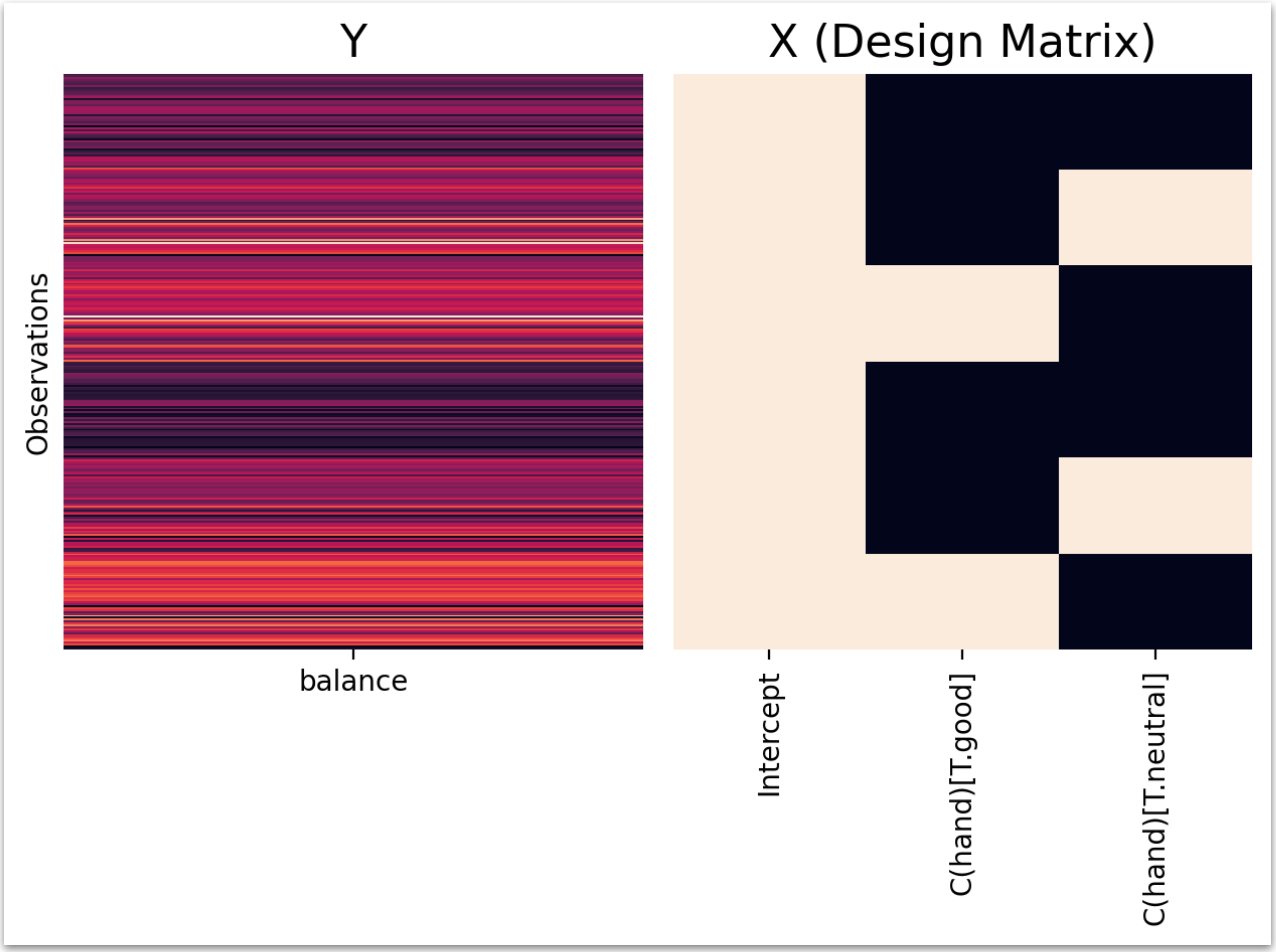
Interpreting parameter estimates

what do these represent?

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                balance   R-squared:                       0.338
Model:                            OLS   Adj. R-squared:                  0.333
Method:                 Least Squares   F-statistic:                     75.70
Date:                Wed, 19 Feb 2025   Prob (F-statistic):           2.70e-27
Time:                        11:57:14   Log-Likelihood:                -848.31
No. Observations:                 300   AIC:                             1703.
Df Residuals:                     297   BIC:                             1714.
Df Model:                           2
Covariance Type:            nonrobust
=========================================================================================
                            coef    std err          t      P>|t|      [0.025      0.975]
-----------------------------------------------------------------------------------------
Intercept                 5.9415      0.411     14.451      0.000       5.132       6.751
C(hand)[T.good]           7.0849      0.581     12.185      0.000       5.941       8.229
C(hand)[T.neutral]        4.4051      0.581      7.576      0.000       3.261       5.549
==============================================================================
```

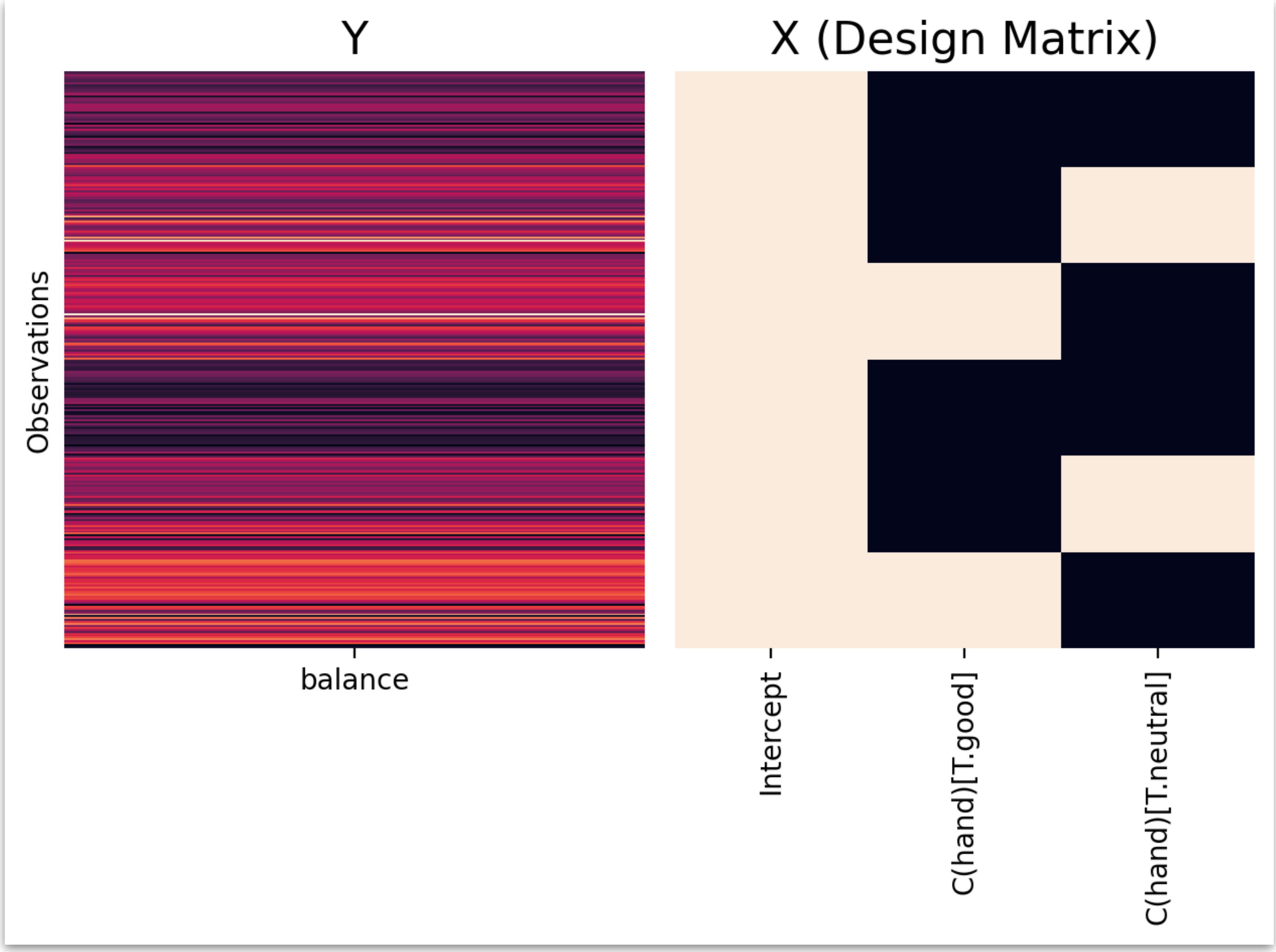
Treatment (Dummy) Coding 3-levels

```python
# Treat "hand" as dummy-coded categorical variable
model_a = ols('balance ~ C(hand)', data=df.to_pandas())
```



Treatment (Dummy) Coding 3-levels

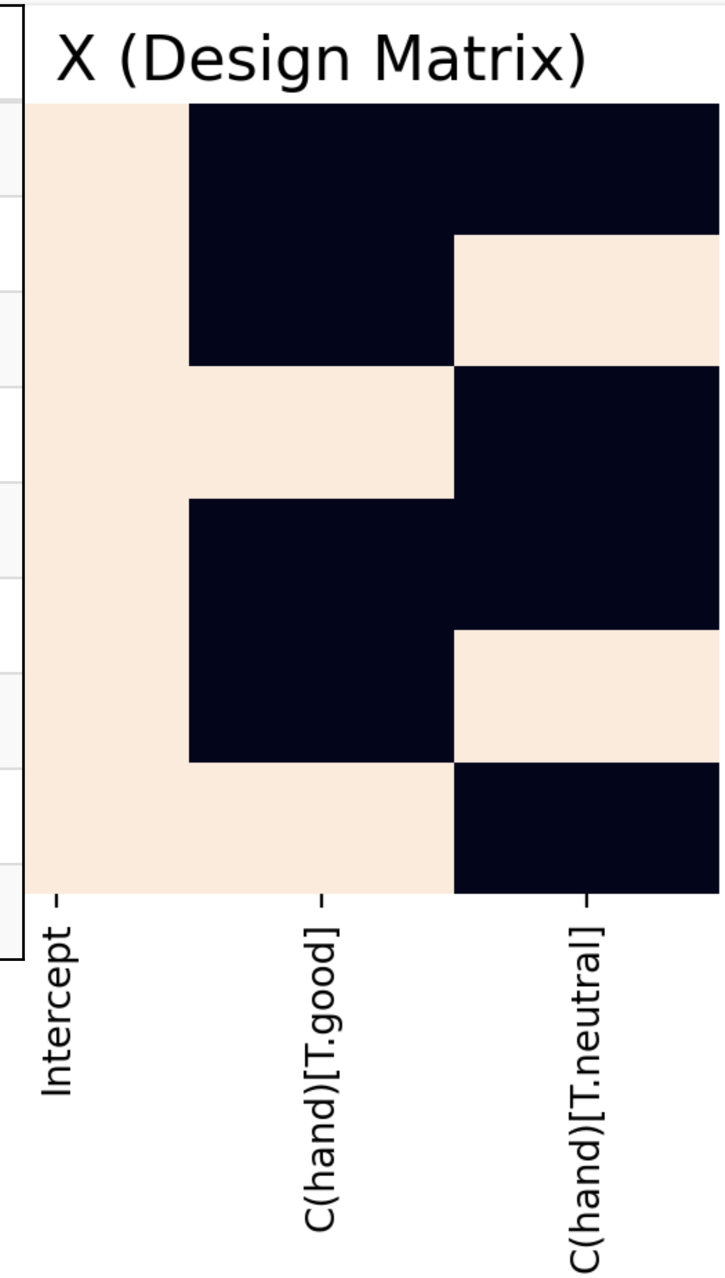
- **Reference level** is coded as 0, and other levels are coded as 1
- **Intercept** = reference (mean); **Slope(s)** = mean difference from reference
- **B0** = ("bad")
- **B1** = ("good" - "bad")
- **B2** = ("neutral" - "bad")



Treatment (Dummy) Coding 3-levels

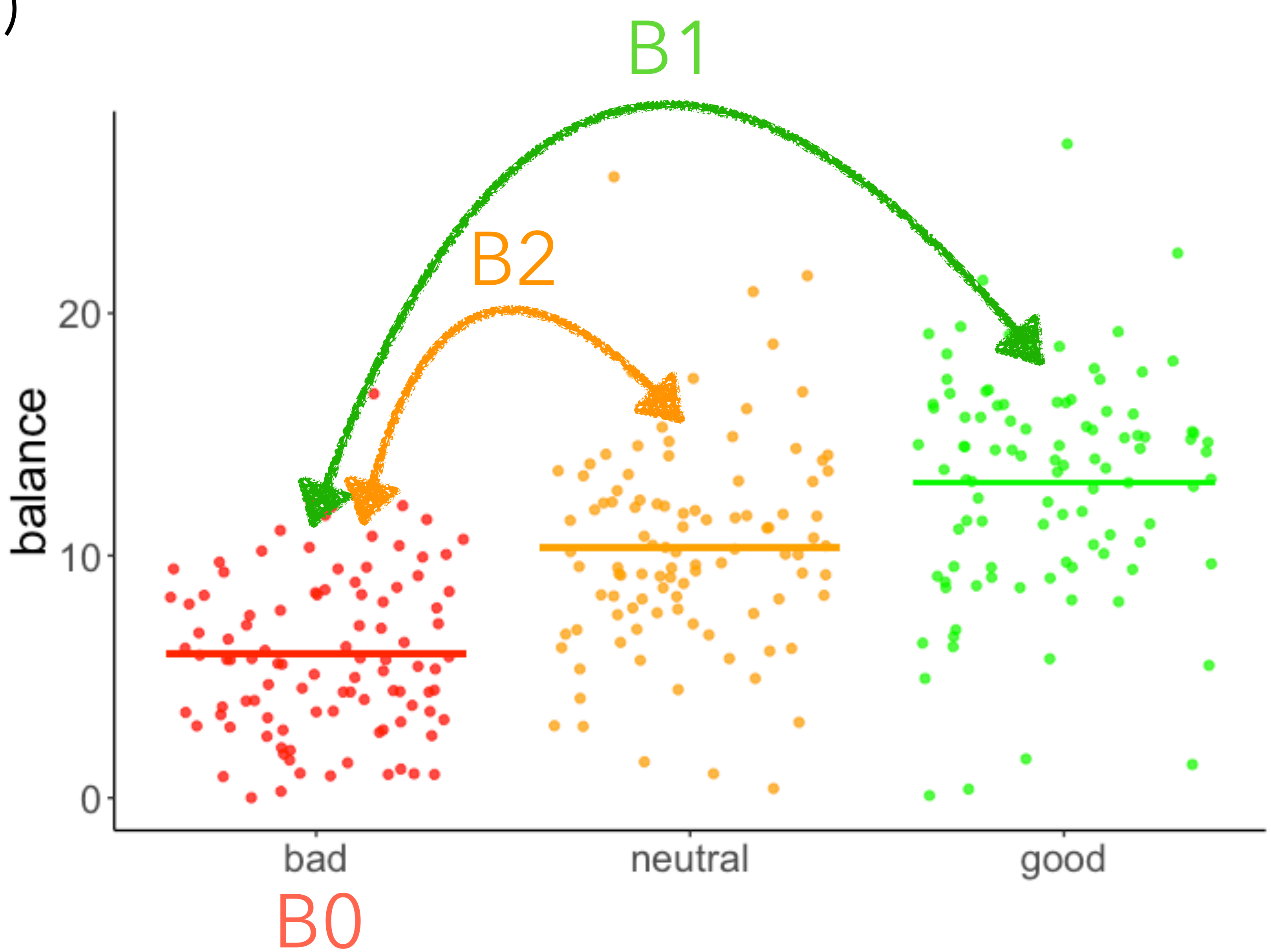
- **Reference level** is coded as 0, and other levels are coded as 1
- **Intercept** = reference (mean); **Slope(s)** = mean difference from reference
- **B0** = (“bad”)
- **B1** = (“good” - “bad”)
- **B2** = (“neutral” - “bad”)

intercept	hand_good	hand_neutral
1	0	0
1	0	0
1	0	0
1	1	0
1	1	0
1	1	0
1	0	1
1	0	1
1	0	1



Treatment (Dummy) Coding 3-levels

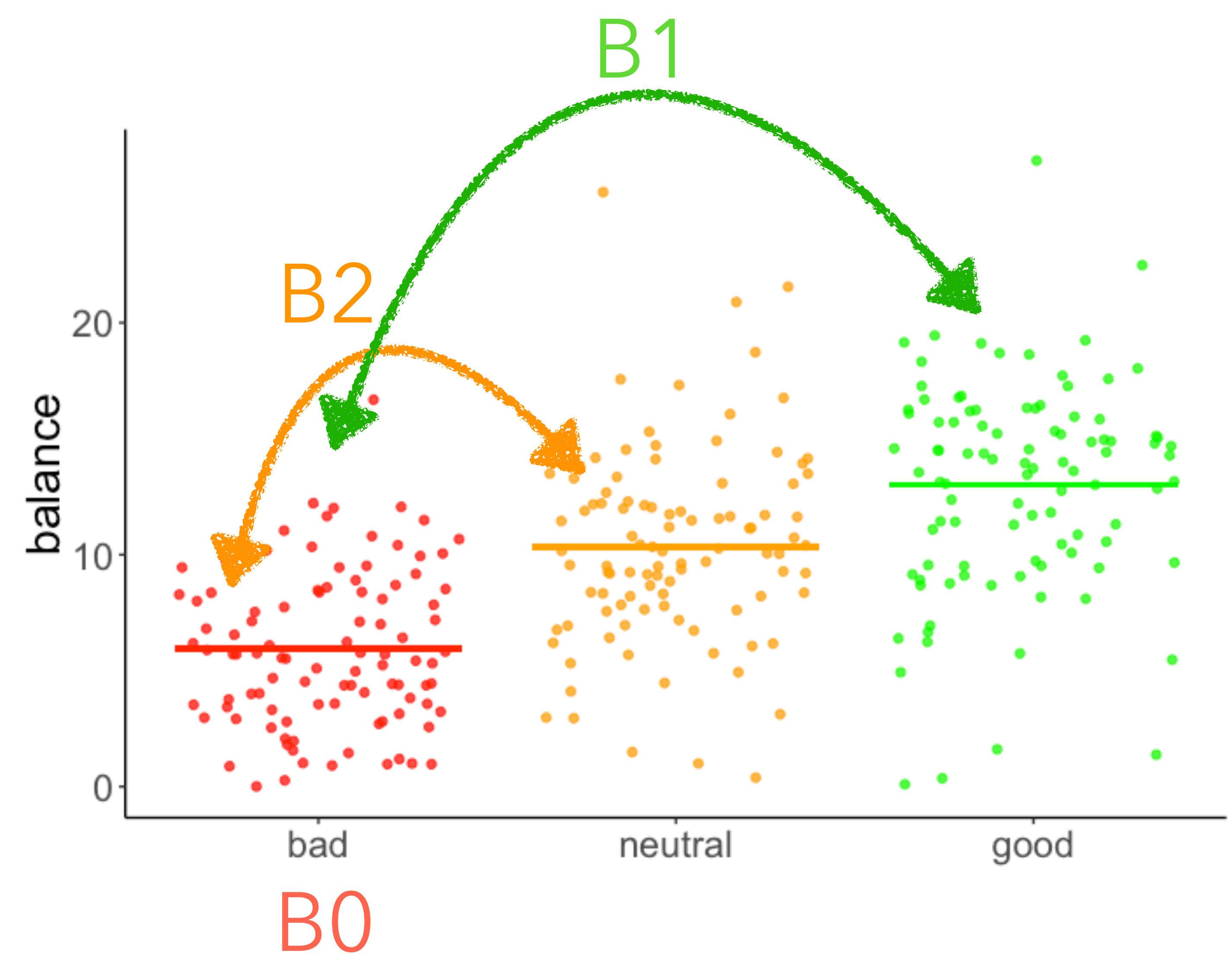
- **Reference level** is coded as 0, and other levels are coded as 1
- **Intercept** = reference (mean); **Slope(s)** = mean difference from reference
- **B0** = (“bad”)
- **B1** = (“good” - “bad”)
- **B2** = (“neutral” - “bad”)



Treatment (Dummy) Coding 3-levels

- **Reference level** is coded as 0, and other levels are coded as 1
- **Intercept** = reference (mean); **Slope(s)** = mean difference from reference

	coef
Intercept	5.9415
C(hand)[T.good]	7.0849
C(hand)[T.neutral]	4.4051



if hand == "bad":

$$\widehat{\text{balance}}_i = 5.94$$

if hand == "good":

$$\widehat{\text{balance}}_i = 5.94 + 7.08 = 13.02$$

if hand == "neutral":

$$\widehat{\text{balance}}_i = 5.94 + 4.41 = 10.35$$

$$\widehat{\text{balance}}_i = 5.94 + 7.08 \cdot \text{hand\_good}_i + 4.41 \cdot \text{hand\_neutral}_i$$

How does the GLM see **categorical variables**?

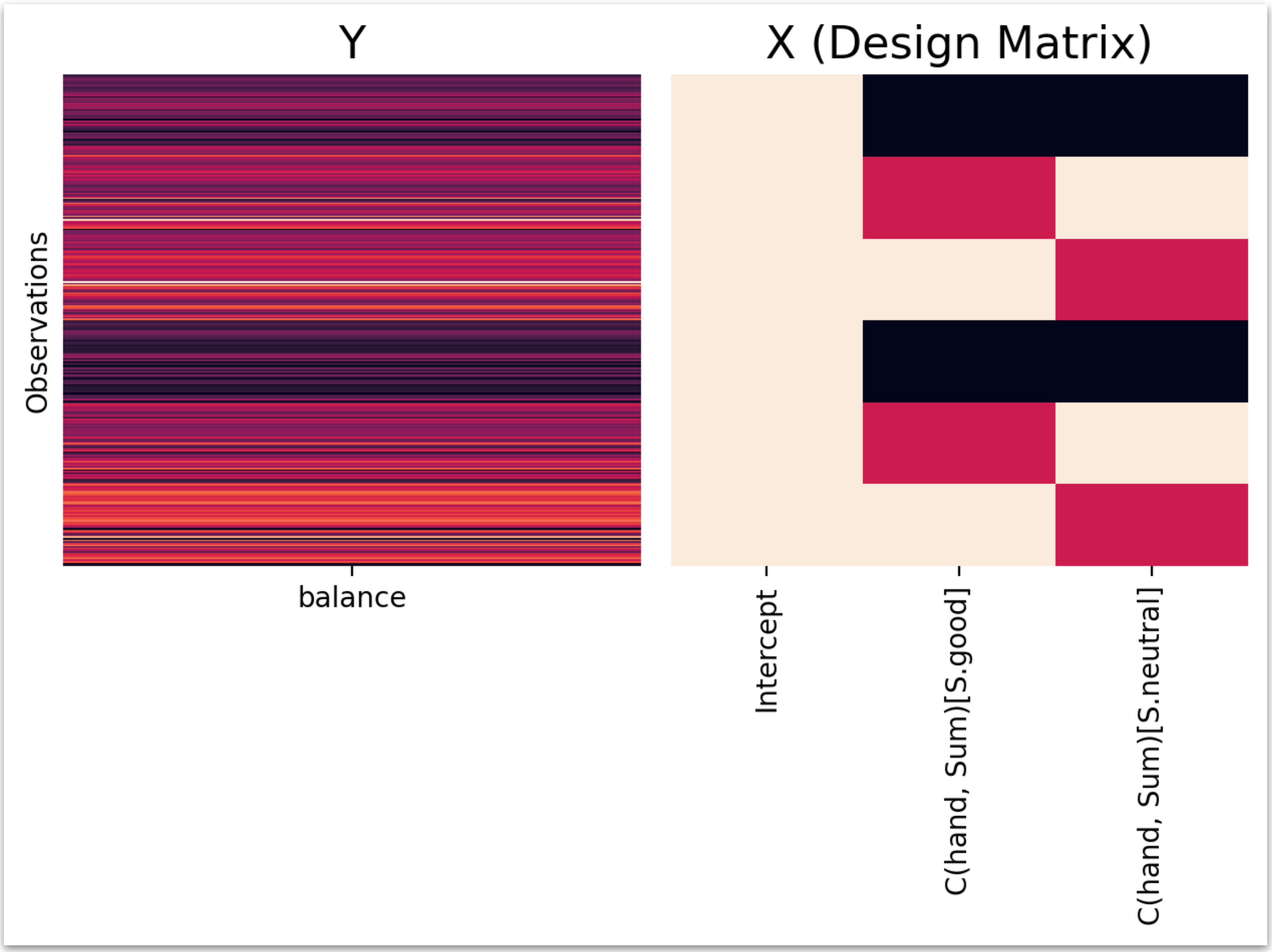
We encode levels of a categorical variable using numbers

We represent ***k levels*** of a categorical variable with ***k-1 parameters*** using one of many possible **coding schemes**

Let's see some more

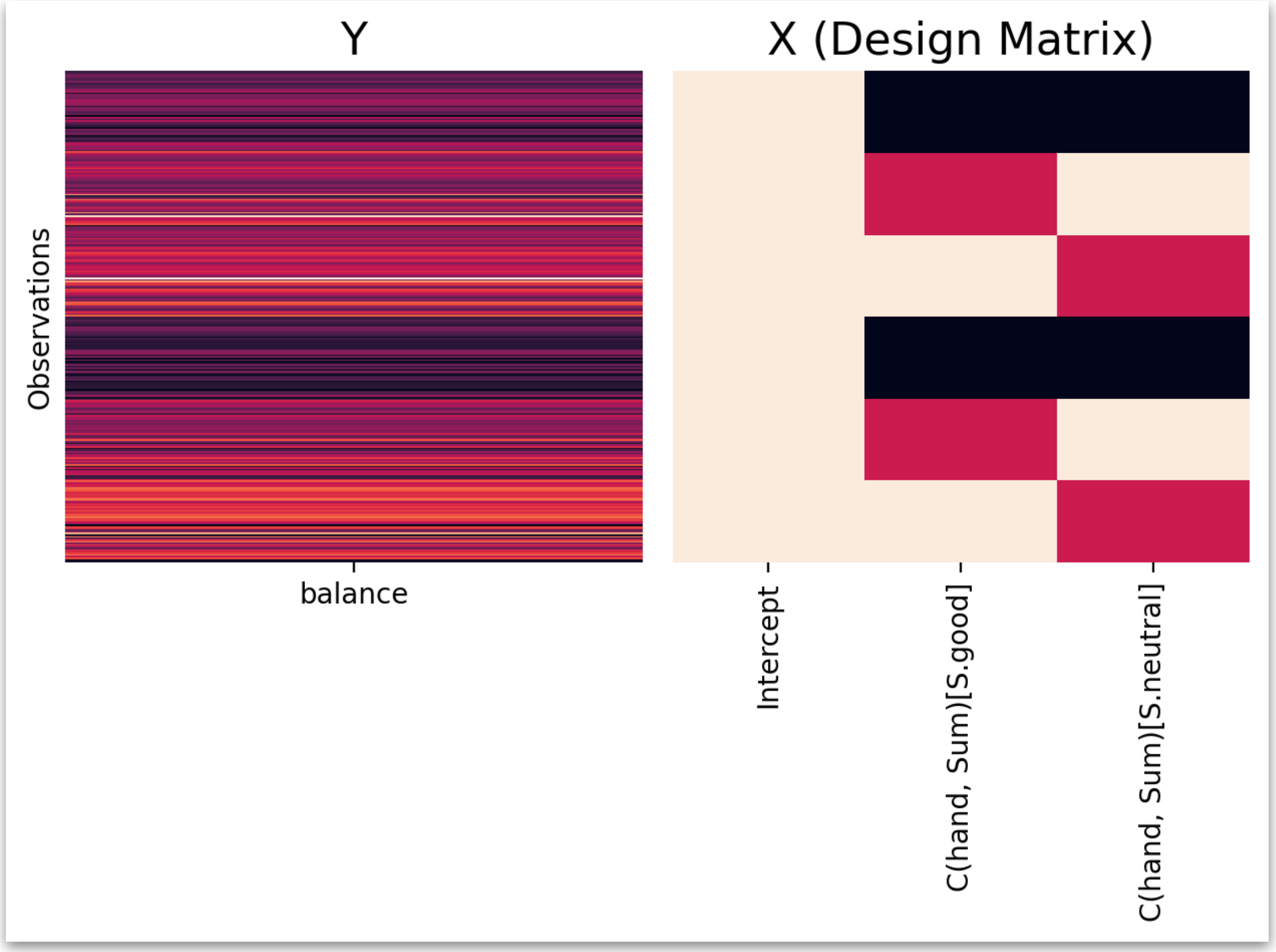
Deviation (Sum/Contrast) Coding

```
# Treat "hand" as sum-coded categorical variable
model_sum = ols("balance ~ C(hand, Sum)", data=df.to_pandas())
```



Deviation (Sum/Contrast) Coding

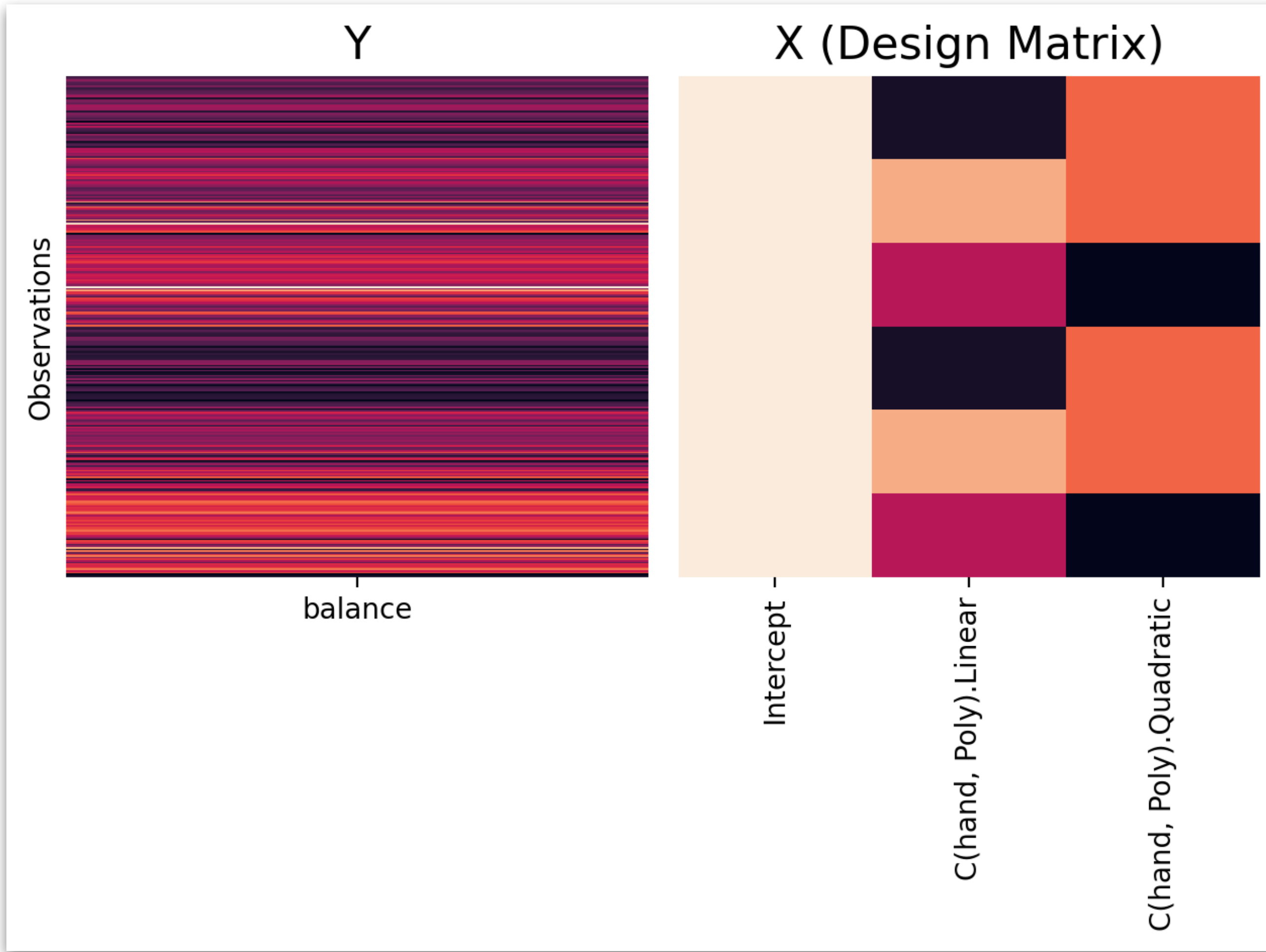
```
# Treat "hand" as sum-coded categorical variable
model_sum = ols("balance ~ C(hand, Sum)", data=df.to_pandas())
```



- Each level is coded as 1; **last level** = -1
- **Intercept** = **grand-mean**; **Slope(s)** = deviations from grand-mean
- Why? You want a valid ANOVA (F-test) and have *at least* 2 predictors
    - At least 1 predictor has 3+ levels

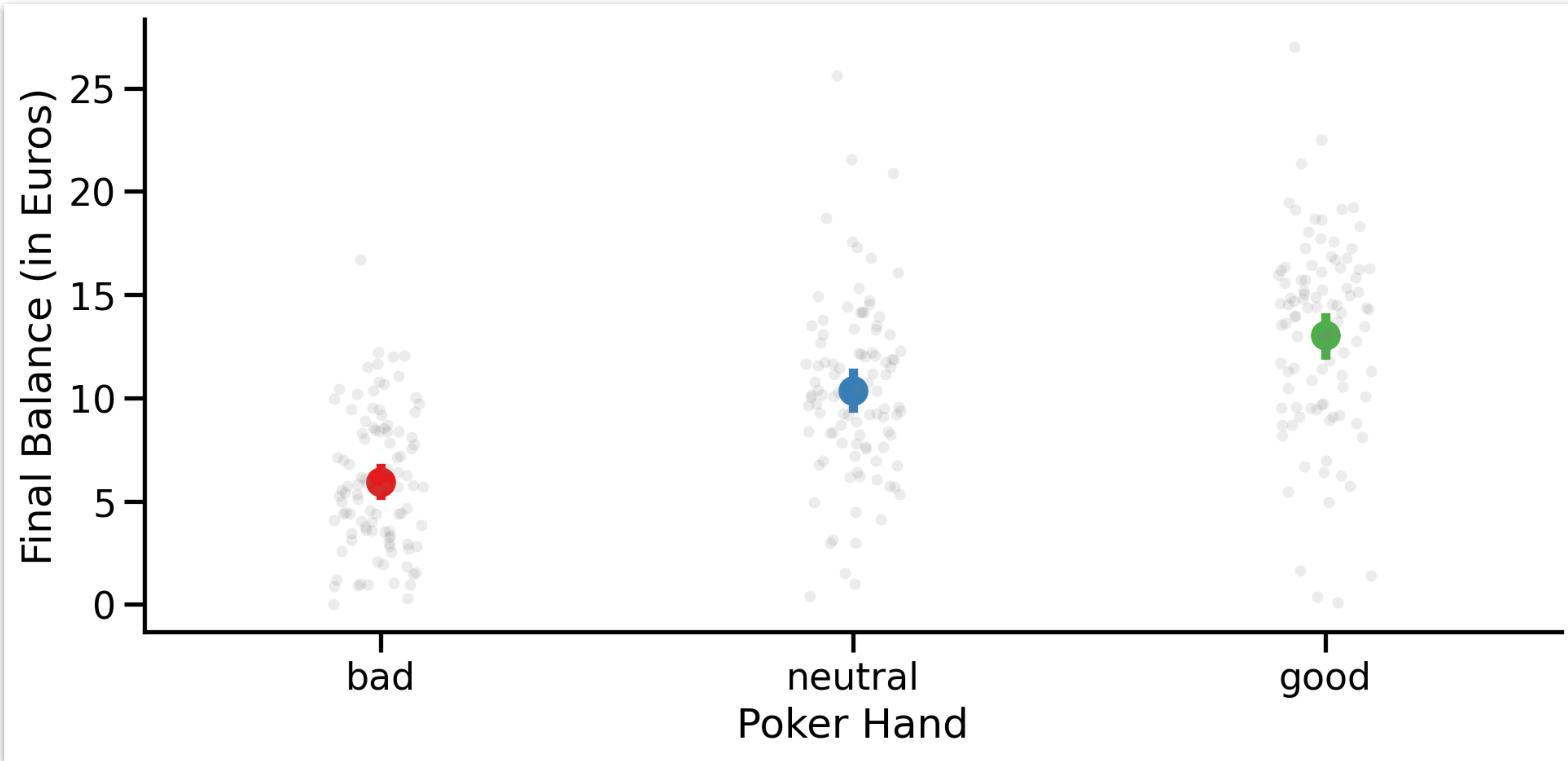
Polynomial (Orthogonal) Coding

```
# Treat "hand" as polynomial-coded categorical variable
model_poly = ols("balance ~ C(hand, Poly)", data=df.to_pandas())
```



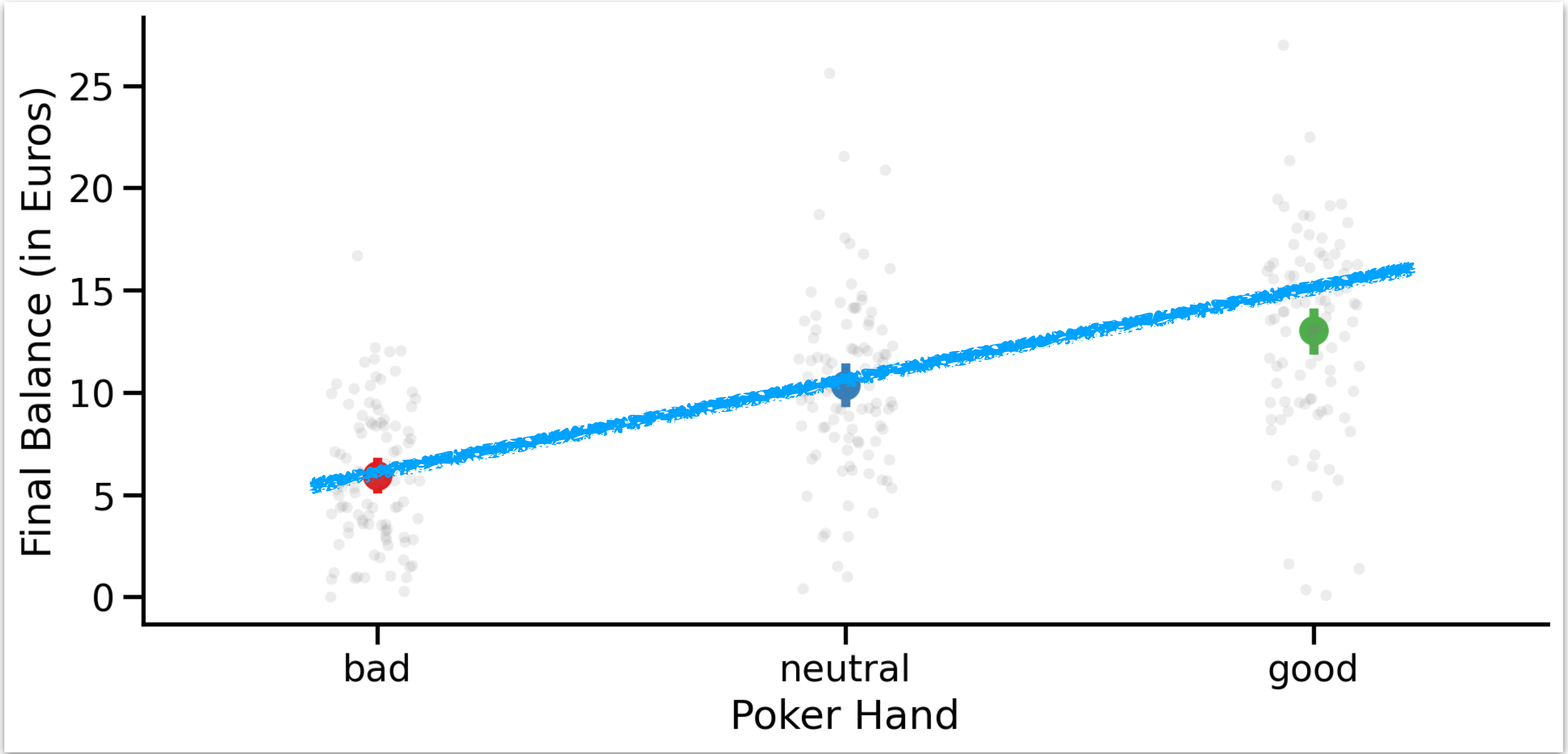
- **Intercept** = **grand-mean**; **Slope(s)** = polynomial (linear, quadratic, cubic..)
- Allows you test specific **trends** over levels of categorical variable
- Why? You want a valid ANOVA (F-test) and have *at least* 2 predictors
  - At least 1 predictor has 3+ levels

Polynomial (Orthogonal) Coding



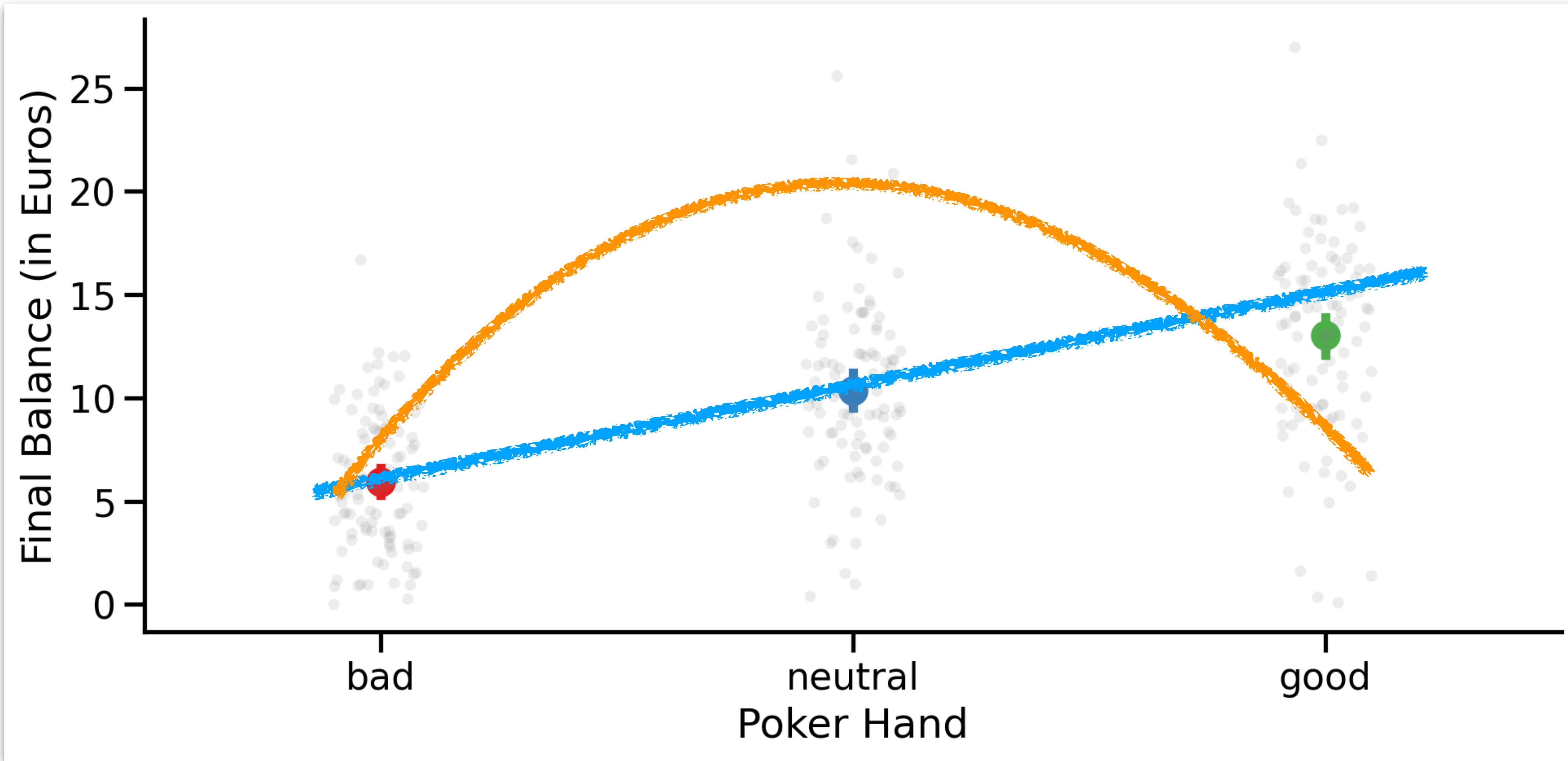
- **Intercept** = **grand-mean**; **Slope(s)** = polynomial (linear, quadratic, cubic..)
- Allows you test specific **trends** over levels of categorical variable
- Why? You want a valid ANOVA (F-test) and have *at least* 2 predictors
  - At least 1 predictor has 3+ levels

Polynomial (Orthogonal) Coding



- **Intercept** = **grand-mean**; **Slope(s)** = polynomial (linear, quadratic, cubic..)
- Allows you test specific **trends** over levels of categorical variable
- Why? You want a valid ANOVA (F-test) and have *at least* 2 predictors
  - At least 1 predictor has 3+ levels

Polynomial (Orthogonal) Coding



- **Intercept** = **grand-mean**; **Slope(s)** = polynomial (linear, quadratic, cubic..)
- Allows you test specific **trends** over levels of categorical variable
- Why? You want a valid ANOVA (F-test) and have *at least* 2 predictors
  - At least 1 predictor has 3+ levels

Today’s Plan

1. First Half (together)
    - Treatment coding review
    - Treatment with 3 levels
2. Second Half (on your own)
    - Notebooks 4, 5, **6, 7**
    - Look at previous notebook solutions if you haven’t