Introduction to Data Science

- More about Multiple Linear Regression & Logistic Regression -

Mauricio Molina

Keio University, Faculty of Economics

Jun 25, 2025

Contents

1. Multiple Linear Regression: Continuation
2. Classification: Logistic Regression
3. Training and Testing a Model
4. Assignment 11

Qualitative Predictors

- Some predictors are not *quantitative* but are *qualitative*, taking a discrete set of values.
- These are also called *categorical* predictors or *factor variables*.
- See for example the scatterplot matrix of the credit card data in the next slide.

In addition to the 7 quantitative variables shown, there are four qualitative variables: `gender`, `student` (student status), `status` (marital status), and `ethnicity` (Caucasian, African American (AA) or Asian).

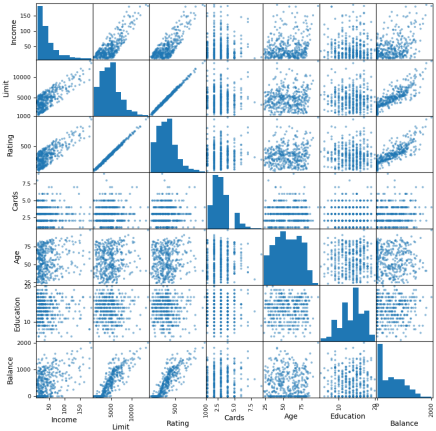


Figure: The `Credit` data set contains information about `balance`, `age`, `cards`, `education`, `income`, `limit`, and `rating` for a number of potential customers.

Example: investigate differences in credit card balance between males and females, ignoring the other variables. We create a new variable **(dummy variable)**

$$x_i = \begin{cases} 1 & \text{if } i \text{ th person is female} \\ 0 & \text{if } i \text{ th person is male} \end{cases}$$

Resulting model:

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i = \begin{cases} \beta_0 + \beta_1 + \epsilon_i & \text{if } i \text{ th person is female} \\ \beta_0 + \epsilon_i & \text{if } i \text{ th person is male} \end{cases}$$

Intrepretation?

Results for gender model:

	Coefficient	Std. Error	t-statistic	p-value
Intercept	509.80	33.13	15.389	< 0.0001
gender[Female]	19.73	46.05	0.429	0.6690

Qualitative predictors with more than two levels

- With more than two levels, we create additional dummy variables. For example, for the ethnicity variable we create two dummy variables. The first could be

$$x_{i1} = \begin{cases} 1 & \text{if } i \text{ th person is Asian} \\ 0 & \text{if } i \text{ th person is not Asian} \end{cases}$$

and the second could be

$$x_{i2} = \begin{cases} 1 & \text{if } i \text{ th person is Caucasian} \\ 0 & \text{if } i \text{ th person is not Caucasian.} \end{cases}$$

Qualitative predictors with more than two levels continued.

- Then both of these variables can be used in the regression equation, in order to obtain the model

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \epsilon_i = \begin{cases} \beta_0 + \beta_1 + \epsilon_i & \text{if } i \text{ th person is Asian} \\ \beta_0 + \beta_2 + \epsilon_i & \text{if } i \text{ th person is Caucasian} \\ \beta_0 + \epsilon_i & \text{if } i \text{ th person is AA.} \end{cases}$$

- There will always be one fewer dummy variable than the number of levels. The level with no dummy variable African American in this example - is known as the baseline.

	Coefficient	Std. Error	t-statistic	p-value
Intercept	531.00	46.32	11.464	< 0.0001
ethnicity [Asian]	-18.69	65.02	-0.287	0.7740
ethnicity [Caucasian]	-12.50	56.68	-0.221	0.8260

Extensions of the Linear Model

Removing the additive assumption: *interactions* and *nonlinearity*.

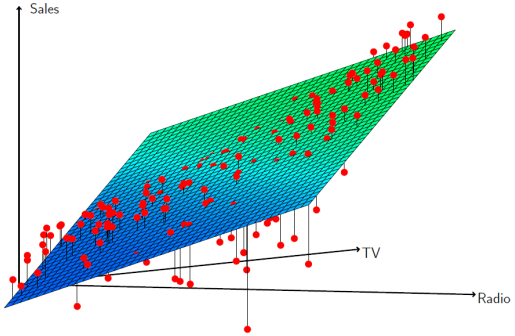
Interactions:

- In our previous analysis of the `Advertising` data, we assumed that the effect on `sales` of increasing one advertising medium is independent of the amount spent on the other media.
- For example, the linear model

$$\widehat{\text{sales}} = \beta_0 + \beta_1 \times \text{TV} + \beta_2 \times \text{ radio } + \beta_3 \times \text{ newspaper}$$

states that the average effect on `sales` of a one-unit increase in TV is always β_1, regardless of the amount spent on radio.

- But suppose that spending money on radio advertising actually increases the effectiveness of TV advertising, so that the slope term for TV should increase as radio increases.
- In this situation, given a fixed budget of $\$100,000$, spending half on radio and half on TV may increase sales more than allocating the entire amount to either TV or to radio.
- In marketing, this is known as a *synergy* effect, and in statistics it is referred to as an *interaction* effect.



When levels of either `TV` or `radio` are low, then the true `sales` are lower than predicted by the linear model.
But when advertising is split between the two media, then the model tends to underestimate `sales`.

Model takes the form

$$\begin{aligned}\texttt{sales} &= \beta_0 + \beta_1 \times \texttt{TV} + \beta_2 \times \texttt{ radio } + \beta_3 \times (\texttt{ radio } \times \texttt{TV}) + \epsilon \\ &= \beta_0 + (\beta_1 + \beta_3 \times \texttt{ radio }) \times \texttt{TV} + \beta_2 \times \texttt{ radio } + \epsilon.\end{aligned}$$

	Coefficient	Std. Error	t-statistic	p-value
Intercept	6.7502	0.248	27.23	< 0.0001
TV	0.0191	0.002	12.70	< 0.0001
radio	0.0289	0.009	3.24	0.0014
TV $\times$ radio	0.0011	0.000	20.73	< 0.0001

Interpretation

- The results in this table suggests that interactions are important.
- The p-value for the interaction term `TV` $\times$ `radio` is extremely low, indicating that there is strong evidence for $H_A : \beta_3 \neq 0$.
- The R^2 for the interaction model is 96.8%, compared to only 89.7% for the model that predicts sales using `TV` and `radio` without an interaction term.

- This means that $(96.8 - 89.7)/(100 - 89.7) = 69\%$ of the variability in sales that remains after fitting the additive model has been explained by the interaction term.
- The coefficient estimates in the table suggest that an increase in TV advertising of $\$1,000$ is associated with increased `sales` of $(\hat{\beta}_1 + \hat{\beta}_3 \times \text{radio}) \times 1000 = 19 + 1.1 \times$ radio units.
- An increase in radio advertising of $\$1,000$ will be associated with an increase in sales of $\left(\hat{\beta}_2 + \hat{\beta}_3 \times \texttt{TV}\right) \times 1000 = 29 + 1.1 \times \texttt{TV}$ units.
- Sometimes it is the case that an interaction term has a very small p-value, but the associated main effects (in this case, `TV` and `radio`) do not.

Interactions between qualitative and quantitative variables

Consider the Credit data set, and suppose that we wish to predict balance using income (quantitative) and student (qualitative).
Without an interaction term, the model takes the form

$$\begin{aligned}\text{balance}_i &\approx \beta_0 + \beta_1 \times \text{income}_i + \begin{cases}\beta_2 & \text{if } i \text{ th person is a student} \\ 0 & \text{if } i \text{ th person is not a student}\end{cases} \\ &= \beta_1 \times \text{income}_i + \begin{cases}\beta_0 + \beta_2 & \text{if } i \text{ th person is a student} \\ \beta_0 & \text{if } i \text{ th person is not a student.}\end{cases}\end{aligned}$$

With interactions, it takes the form

$$\text{balance}_i \approx \beta_0 + \beta_1 \times \text{income}_i + \begin{cases} \beta_2 + \beta_3 \times \text{income}_i & \text{if student} \\ 0 & \text{if not student} \end{cases}$$

$$= \begin{cases} (\beta_0 + \beta_2) + (\beta_1 + \beta_3) \times \text{income}_i & \text{if student} \\ \beta_0 + \beta_1 \times \text{income}_i & \text{if not student} \end{cases}$$

Non-linear Relationships

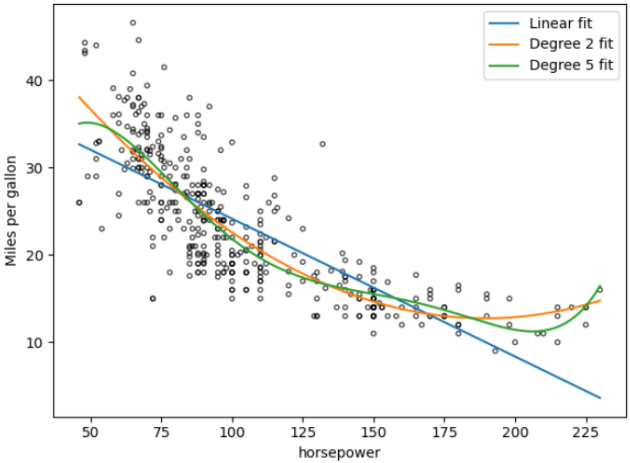


Figure: Polynomial regression on Auto data

The figure suggests that

$$\texttt{mpg} = \beta_0 + \beta_1 \times \texttt{ horsepower } + \beta_2 \times \texttt{ horsepower }^2 + \epsilon$$

may provide a better fit.

	Coefficient	Std. Error	t-statistic	p-value
Intercept	56.9001	1.8004	31.6	< 0.0001
horsepower	-0.4662	0.0311	-15.0	< 0.0001
horsepower ^{2}	0.0012	0.0001	10.1	< 0.0001

Classification

- Qualitative variables take values in an unordered set $\mathcal{C}$, such as:
  `eye color` $\in \{$ `brown`, `blue`, `green` $\}$
  email $\in \{$`spam`, `ham` $\}$.
- Given a feature vector X and a qualitative response Y taking values in the set $\mathcal{C}$, the classification task is to build a function $C(X)$ that takes as input the feature vector X and predicts its value for Y; i.e. $C(X) \in \mathcal{C}$.
- Often we are more interested in estimating the probabilities that X belongs to each category in $\mathcal{C}$.

For example, it is more valuable to have an estimate of the probability that an insurance claim is fraudulent, than a classification fraudulent or not.

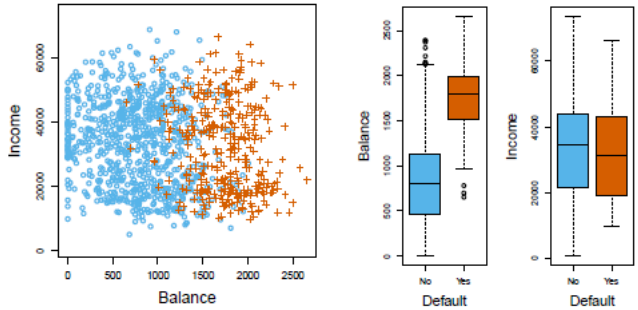


Figure: *The* `Default` *data set. Left: The annual incomes and monthly credit card balances of a number of individuals. The individuals who defaulted on their credit card payments are shown in orange, and those who did not are shown in blue. Center: Boxplots of* `balance` *as a function of* `default` *status. Right: Boxplots of* `income` *as a function of* `default` *status.*

Can we use Linear Regression?

Suppose for the `Default` classification task that we code

$$Y = \begin{cases} 0 & \text{if } \texttt{No} \\ 1 & \text{if } \texttt{Yes} \end{cases}$$

Can we simply perform a linear regression of Y on X and classify as `Yes` if $\hat{Y} > 0.5$?

- In this case of a binary outcome, linear regression does a good job as a classifier.
- Since in the population $E(Y \mid X = x) = \Pr(Y = 1 \mid X = x)$, we might think that regression is perfect for this task.
- However, linear regression might produce probabilities less than zero or bigger than one. *Logistic regression* is more appropriate.

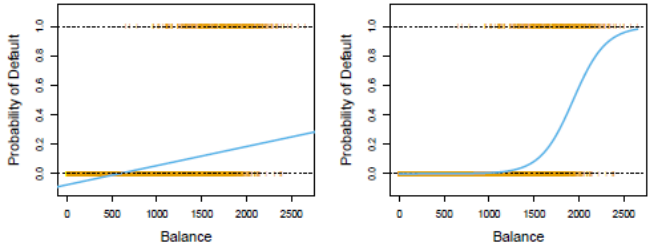


Figure: *Classification using the `Default` data. Left: Estimated probability of default using linear regression. Some estimated probabilities are negative! The orange ticks indicate the 0/1 values coded for `default`(`No` or `Yes`). Right: Predicted probabilities of `default` using logistic regression. All probabilities lie between 0 and 1.*

Now suppose we have a response variable with three possible values. A patient presents at the emergency room, and we must classify them according to their symptoms.

$$Y = \begin{cases} 1 & \text{if } \texttt{stroke}; \\ 2 & \text{if drug } \texttt{overdose}; \\ 3 & \text{if } \texttt{epileptic seizure}. \end{cases}$$

This coding suggests an ordering, and in fact implies that the difference between `stroke` and `drug overdose` is the same as between `drug overdose` and `epileptic seizure`.
Linear regression is not appropriate here.
Multiclass Logistic Regression or *Discriminant Analysis* are more appropriate.

Logistic Regression

Let's write $p(X) = \Pr(Y = 1 \mid X)$ for short and consider using `balance` to predict `default`. Logistic regression uses the form

$$p(X) = \frac{e^{\beta_0+\beta_1 X}}{1 + e^{\beta_0+\beta_1 X}}$$

($e \approx 2.71828$ is a mathematical constant [Euler's number.])
It is easy to see that no matter what values β_0, β_1 or X take, $p(X)$ will have values between 0 and 1 .
A bit of rearrangement gives

$$\log\left(\frac{p(X)}{1 - p(X)}\right) = \beta_0 + \beta_1 X$$

This monotone transformation is called the *log odds* or *logit* transformation of $p(X)$. (by log we mean natural log: ln.)

Maximum Likelihood

We use maximum likelihood to estimate the parameters.

$$\ell(\beta_0, \beta) = \prod_{i:y_i=1} p(x_i) \prod_{i:y_i=0} (1 - p(x_i))$$

This likelihood gives the probability of the observed zeros and ones in the data. We pick β_0 and β_1 to maximize the likelihood of the observed data.
Most statistical packages can fit linear logistic regression models by maximum likelihood. In Python, using the `statsmodels.api`, we use the `Logit` function.

	Coefficient	Std. Error	Z-statistic	P-value
Intercept	-10.6513	0.3612	-29.5	< 0.0001
balance	0.0055	0.0002	24.9	< 0.0001

Making Predictions

What is our estimated probability of default for someone with a balance of \$1000 ?

$$\hat{p}(X) = \frac{e^{\hat{\beta}_0+\hat{\beta}_1 X}}{1+e^{\hat{\beta}_0+\hat{\beta}_1 X}} = \frac{e^{-10.6513+0.0055\times 1000}}{1+e^{-10.6513+0.0055\times 1000}} = 0.006$$

With a balance of \$2000 ?

$$\hat{p}(X) = \frac{e^{\hat{\beta}_0+\hat{\beta}_1 X}}{1+e^{\hat{\beta}_0+\hat{\beta}_1 X}} = \frac{e^{-10.6513+0.0055\times 2000}}{1+e^{-10.6513+0.0055\times 2000}} = 0.586$$

Lets do it again, using student as the predictor.

	Coefficient	Std. Error	Z-statistic	P-value
Intercept	-3.5041	0.0707	-49.55	< 0.0001
student [Yes]	0.4049	0.1150	3.52	0.0004

$$\widehat{\mathrm{P}}(\texttt{default} = \texttt{Yes} \mid \texttt{student} = \texttt{Yes}) = \frac{e^{-3.5041+0.4049\times 1}}{1+e^{-3.5041+0.4049\times 1}} = 0.0431$$

$$\widehat{\mathrm{P}}(\texttt{default} = \texttt{Yes} \mid \texttt{student} = \texttt{No}) = \frac{e^{-3.5041+0.4049\times 0}}{1+e^{-3.5041+0.4049\times 0}} = 0.0292$$

Logistic regression with several variables

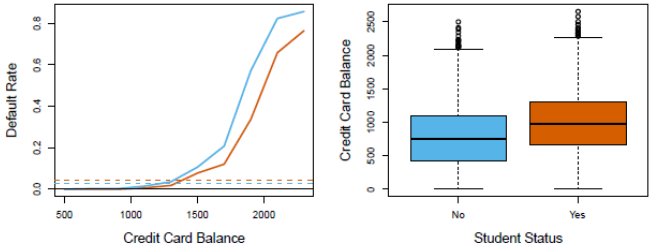
$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$$

$$p(X) = \frac{e^{\beta_0+\beta_1 X_1+\cdots+\beta_p X_p}}{1+e^{\beta_0+\beta_1 X_1+\cdots+\beta_p X_p}}$$

	Coefficient	Std. Error	Z-statistic	P-value
Intercept	-10.8690	0.4923	-22.08	< 0.0001
balance	0.0057	0.0002	24.74	< 0.0001
income	0.0030	0.0082	0.37	0.7115
student [Yes]	-0.6468	0.2362	-2.74	0.0062

Why is coefficient for `student` negative, while it was positive before?

Confounding



- Students tend to have higher balances than non-students, so their marginal default rate is higher than for non-students.
- But for each level of balance, students default less than non-students.
- Multiple logistic regression can tease this out.

Why Split Data into Train and Test Sets?

- **Estimate true performance** Testing on held-out data reveals generalization error, not just training fit.
- **Avoid optimistic bias** Evaluating on training data alone overestimates accuracy (or R^2).
- **Detect overfitting** A large gap between train and test scores signals a model that has memorized noise.
- **Reliable model selection** Compare different algorithms or hyper-parameters on the same unseen test set.

Introducing scikit-learn's `model_selection`

- `train_test_split`:
  - Splits arrays or DataFrames into random train and test subsets.
  - Syntax: `X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)`
- `cross_val_score`:
  - Performs k-fold cross-validation easily.
  - Syntax: `scores = cross_val_score(model, X, y, cv=5)`
- Consistent API for regression, classification, clustering, etc.

Overfitting and the `score()` Method

Overfitting happens when a model learns random quirks in the training data instead of the true underlying patterns. Such a model may score perfectly on known data but will usually perform poorly on new data, defeating its predictive purpose. By avoiding overfitting—through practices like hold-out testing, cross-validation, or regularization—you build simpler, more robust models that generalize well and yield trustworthy insights.

- **Overfitting:** Model fits training data too closely, capturing noise instead of signal.
- **Symptoms:** $\text{score}_{\text{train}} \gg \text{score}_{\text{test}}$
- `model.score(X, y)`
  - *Regression:* Returns R^2 on (X, y).
  - *Classification:* Returns accuracy on (X, y).
- Always compare `score()` on train vs. test to gauge overfitting.

Step-by-Step: Build & Validate a Model

1. **Import libraries:** `from sklearn.model_selection import train_test_split from sklearn.linear_model import LinearRegression` (or `LogisticRegression`)
2. **Load and preprocess data:** Drop NaN, encode categoricals, scale if needed.
3. **Split:** `X_train, X_test, y_train, y_test = train_test_split(...)`
4. **Instantiate model:** `model = LinearRegression()` / `LogisticRegression()`
5. **Fit on training set:** `model.fit(X_train, y_train)`
6. **Evaluate:** `train_score = model.score(X_train, y_train)` `test_score = model.score(X_test, y_test)`

Assignment 11

Answer all the following questions in the Jupyter Notebook format. Show your Python code (when requested) and a short explanation for every result. Upload the completed `.ipynb` to K-LMS by next Tuesday at midnight.

Q1:

1. Using Python (`pandas` and `statsmodels`):
   - Load `Credit` into a DataFrame and display the first 5 rows.
   - Create dummy columns: `status_Single` (1 if status=="Single" else 0), `ethnicity_Asian`, `ethnicity_Caucasian` (baseline: African American).
   - Fit OLS regression:

$$\begin{aligned}\texttt{balance} =& \beta_0 + \beta_1 \times \texttt{income} + \beta_2 \times \texttt{status\_Single} + \\ & \beta_3 \times \texttt{ethnicity\_Asian} + \beta_4 \times \texttt{ethnicity\_Caucasian} + \varepsilon.\end{aligned}$$

   - Print the summary table, report each coefficient, its p-value, and indicate which predictors are significant at 5%.
   - Add interaction `income` × `status_Single`, refit, and explain how it changes the income slope for single customers.

Q2:

1. Load `Auto` data and drop any missing values.
2. Define three models:
   - Model A: $\texttt{mpg} \sim \beta_0 + \beta_1 \times \texttt{horsepower}$.
   - Model B: $\texttt{mpg} \sim \beta_0 + \beta_1 \times \texttt{horsepower} + \beta_2 \times \texttt{horsepower}^2$.
   - Model C: $\texttt{mpg} \sim \beta_0 + \sum_{k=1}^{5} \beta_k \times \texttt{horsepower}^k$.
3. Split with `train_test_split(X, y, test_size=0.5, random_state=2025)`.
4. For each model, fit on training data, compute `model.score(X_train, y_train)` and `model.score(X_test, y_test)`.
5. Present training and test R^2 scores in a table and include a plot of the scatter plus all three fitted curves.
6. Discuss which degrees show signs of overfitting based on the score gap.

Q3:

1. Load `Default` data into a DataFrame and encode:
   - `default_bool`: 1 if default=="Yes", else 0.
   - `student_bool`: 1 if student=="Yes", else 0.
2. For each of the three test sizes (0.3, 0.5, 0.7):
   - Split with `train_test_split(X, y, test_size=test_size, random_state=0)`.
   - Fit logistic regression:

$$\Pr(\texttt{default} = 1) = \mathrm{Logit}(\beta_0 + \beta_1\,\texttt{balance} + \beta_2\,\texttt{income} + \beta_3\,\texttt{student\_bool}).$$

   - Compute train and test accuracy via `model.score(...)`.
3. Summarize train/test accuracy in a table.
4. Plot test accuracy vs. test_size and comment on how training size affects performance.