2143488 BIG DATA AND ARTIFICIAL INTELLIGENCE

LOGISTIC REGRESSION

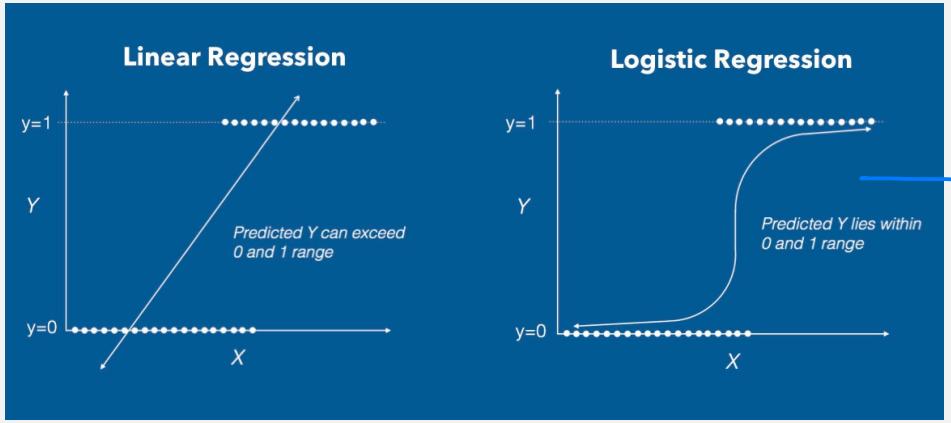
DR. JING TANG

LINEAR REGRESSION CANNOT SOLVE CLASSIFICATION

- If the label/target y is a **binary response** or a **categorical response**, it is not suitable to use linear regression (LR).
  - Using LR for classification violates statistical assumptions.
  - LR minimizes SSE, but not misclassified cases.
  - Probabilities do not work as well with a straight line as they do with a sigmoid curve.

LOGISTIC REGRESSION

- Name is somewhat misleading. Really a technique for **classification**, not regression.
  - "Regression" comes from fact that we fit a linear model to the feature space.
- Involves a more **probabilistic** view of classification.



CDF

→ Sigmoid line

in case there are multiple types

CLASSIFICATION

- **Learn**: h: $\boldsymbol{X} \rightarrow y$
  - $\boldsymbol{X}$ – features , ind
  - y – target class (i.e., “spam”) label . dep
- Suppose you know probability $p(y|\boldsymbol{X})$ exactly, how should you classify?

$p(y_0|X)$ $p(y_1|X)$ $p(y_2|X)$

DIFFERENT WAYS OF EXPRESSING PROBABILITY

- Consider Y as a binominal variable, where:
  - $P(Y = 1) = p$
  - $P(Y = 0) = 1 - p = q$
- Can express probability of Y as:

	notation	range equivalents		
standard probability	p	0	0.5	1
odds	p / q	0	1	+ ∞
log odds (logit)	log(p / q)	- ∞	0	+ ∞

LOG ODDS

- $Log\ odds\ (odds\ of\ success) = \log\left(\frac{p}{1-p}\right)$
  ↳ not % of success !
- Numeric treatment of outcome $\mathrm{Y} = \mathrm{y}$ is equivalent
  - If neither outcome is favored over the other, then $\log odds = 0$.
  - If one outcome is favored with $\log odds > 0$, then the other outcome is disfavored with $\log odds < 0$.

FROM PROBABILITY TO LOG ODDS (AND BACK AGAIN)

$$z = \log\left(\frac{p}{1-p}\right) \qquad \text{logit function}$$

$$\frac{p}{1-p} = e^z$$

$$p = \frac{e^z}{1+e^z} = \frac{1}{1+e^{-z}} \qquad \text{logistic function}$$

SCENARIO OF LOGISTIC REGRESSION

- A multidimensional feature space (feature X_i can be categorical or continuous).
- Outcome Y is **discrete** not continuous.
  - We'll focus on case of two classes, then Y is a binary
- It seems plausible that a linear decision boundary (hyperplane) will give good predictive accuracy.

USING A LOGISTIC REGRESSION MODEL

- Model consists of a vector $\vec{\beta}$ in n-dimensional feature space
- For a point $\vec{x}$ in feature space, project it onto β to convert it into a real number it into a real number z in the range in the range -∞ to +∞.

$$\log\left(\frac{p}{1-p}\right) = z = \alpha + \vec{\beta} \cdot \vec{x} = \alpha + \beta_1 x_1 + \cdots + \beta_d x_d$$

$$\frac{p}{1-p} = e^{\alpha+\beta_1 x_1+\cdots+\beta_d x_d} \Rightarrow x_i \; increases \; 1, then \; \frac{p}{1-p} \; increases \; e^{\beta_i}$$

$$p = \frac{1}{(1+e^{-z})}$$

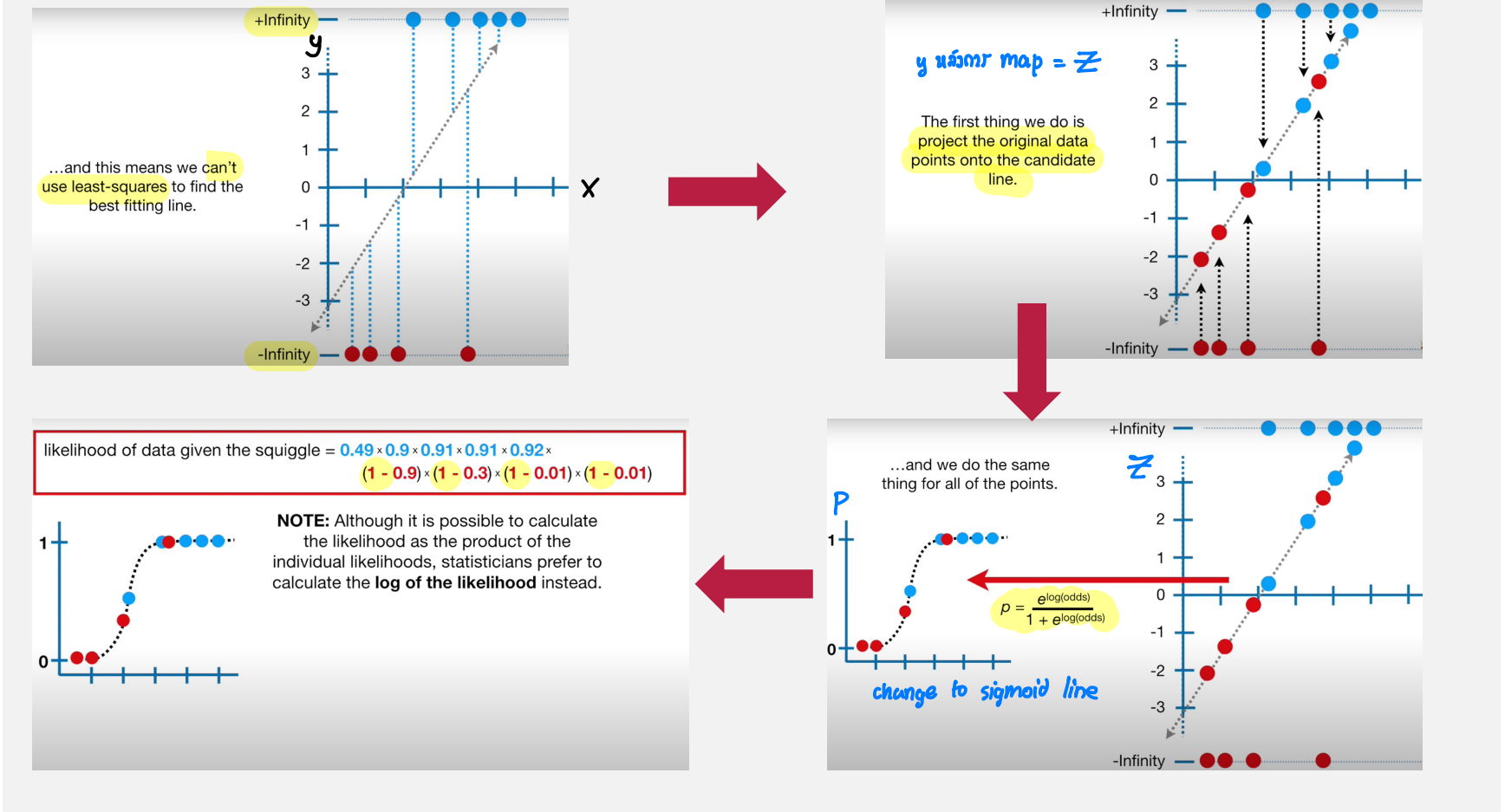
Calculate p by z

- Overall, logistic regression **maps a point** $\vec{x}$ in n-dimensional feature space **to a probability value** in the range 0 to 1

TRAINING A LOGISTIC REGRESSION MODEL

- Need to optimize $\vec{\beta}$ so the model gives the best possible reproduction of training set labels
  - Usually done by numerical approximation of **maximum likelihood** common for classification
    - Max $Joint\ probability\ of\ all\ points = \prod_i p_i$
    - Max $\log(Joint\ probability\ of\ all\ point) = \log(\prod_i p_i) = \sum_i \log P_i$
  - On really large datasets, may use stochastic gradient descent

MAXIMUM LIKELIHOOD (1)



MAXIMUM LIKELIHOOD (2)

likelihood of data given the squiggle = 0.49 × 0.9 × 0.91 × 0.91 × 0.92 × (1 - 0.9) × (1 - 0.3) × (1 - 0.01) × (1 - 0.01)

log(likelihood of data given the squiggle) = log(0.49) + log(0.9) + log(0.91) + log(0.91) + log(0.92) + log(1 - 0.9) + log(1 - 0.3) + log(1 - 0.01) + log(1 - 0.01)

$$\log(likelihood\ of\ data\ given\ the\ squiggle) = -1.64$$

Next, Rotate the line, until find the best value for log(likelihood of data given the squiggle)!!

ADVANTAGES AND DISADVANTAGES

PROS

- Makes **no assumptions about distributions** of classes in feature space
- Easily extend to **multiple classes**
  - p_0 as a base, predict $\log\left(\frac{p_{i\neq 0}}{p_0}\right)$ compare $p_i \neq 0$ to p_0
  - use zero as a base
- Quick to train and predict
- Good accuracy for many simple data sets ↳ not many ✕
- Less inclined to overfitting by using L_2 regularization for imbalanced dataset
  - ↳ 0 category : male 90%, female 10% (imbalanced data set)
- Can interpret model coefficients

CONS

- # of rows (recaords) > # of cols (features)
- Linear decision boundary
- Non-linear problem cannot be solved

$\log\left(\frac{P_1}{P_0}\right) > 0$ $\log\left(\frac{P_2}{P_0}\right) > 0$

which class will you chose?

↳ choose the larger one. P is much bigger than P_0 = better.

if $P_1 > 0$, $P_2 < 0$ → choose P_1

if $P_1 < 0$, $P_2 < 0$ → choose P_0

ASSUMPTIONS OF LOGISTIC REGRESSION

- Linear relationship between the features and log odds of the label
- No high intercorrelations (i.e., multicollinearity) among features
- There are no influential values (extreme values or outliers)
- Independence of errors or no significant autocorrelation
- Sample size is large (>50 sample / feature)

PROCEDURE 1

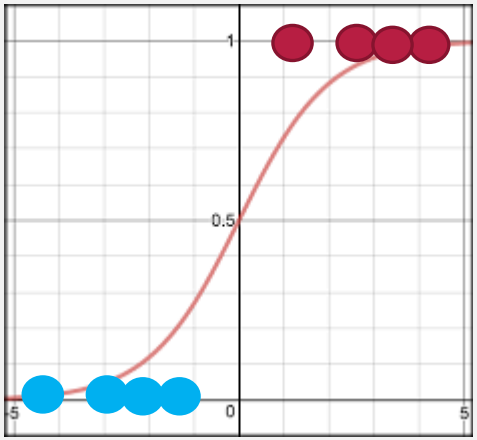
- Load data: Merge & Concatenation
- Explore data: “Garbage in, garbage out”
  - Size (col & row) :
    - At least an order of magnitude more examples than trainable parameters
    - **Simple models on large data sets** generally BEAT fancy models on small data sets
  - Quality
    - Realiability: null, duplication, bad label, bad feature value
    - Feature Representation
    - Distribution

PROCEDURE 2

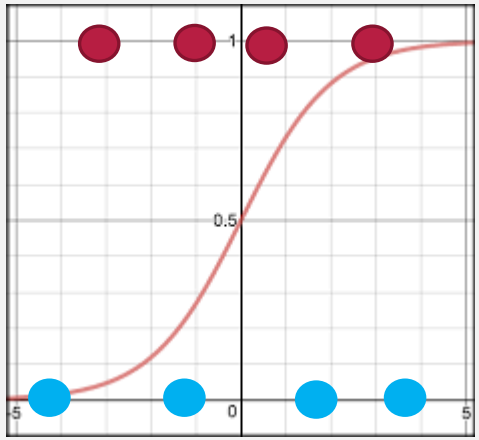
- Preprocess data:
  - Split train-test and put the test data aside
  - Encoding features:
    - Integer → Float; Categorical → Binary Values
  - Fill in missing value in train dataset
  - Scale all features into comparable ranges in train dataset
  - Select features (only numerical) in train dataset
- Model:
  - Debug Logistic model to make the model work
  - Preprocess (based on train dataset) and predict test dataset
  - Evaluate classification result of test dataset
  - Interpret the Logistic Regression function

EXE. WHICH ONE IS MORE SUITABLE FOR LOGISTIC REGRESSION

SCATTER PLOT OF Y and X_1



SCATTER PLOT OF Y and X_2



True
Positive : 1 Negative = 0

Predict Positive = 1 / Negative = 0

True Positive	False Positive
False Negative	True Negative

CLASSIFICATION EVALUATIONS 1

- $\text{Accuracy} = \frac{\text{\# of curret predictions}}{\text{\# of total predictions}} = \frac{TP + TN}{TP + FP + TN + FN}$
  - Not good for imbalanced data examples, with skewed class proportions
    - 99% of website visitors don't buy and that only 1% of visitors buy something
  - Majority vs. Minority classes

Degree of imbalance	Proportion of Minority Class
Mild	20-40% of the data set
Moderate	1-20% of the data set
Extreme	<1% of the data set

CLASSIFICATION EVALUATIONS 2

- $\text{Precision} = \frac{\#\text{ of true positives}}{\#\text{ of true positives} + \#\text{ of false positives}}$
  - Within everything that has been predicted as a positive, precision counts the percentage that is correct
- $\text{Recall} = \frac{\#\text{ of true positives}}{\#\text{ of true positives} + \#\text{ of false negativess}}$
  - Within everything that actually is positive, how many did the model succeed to find

CLASSIFICATION EVALUATIONS 3

- Precision vs. Recall
  - Care only one:
    - i.e., if a shop is interested in making sure that they find all the problematic products back. It does not really matter to them if clients send back some non-problematic products as well, so the precision is not of interest to this supermarket.
  - Care both:
    - Precision-Recall Trade-Off

CLASSIFICATION EVALUATIONS 4

- $\text{F1 score} = 2 \times \frac{Precision \times Recall}{Precision + Recall}$
  - The F1 score is defined as the harmonic **mean of precision and recall**
  - A model will obtain a **high F1 score** if both Precision and Recall are high
  - A model will obtain a **low F1 score** if both Precision and Recall are low
  - A model will obtain a **medium F1 score** if one of Precision and Recall is low and the other is high

CLASSIFICATION EVALUATIONS 5

- **ROC curve** (**receiver operating characteristic curve**):a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters:
  - **True Positive Rate** (**TPR**) is a synonym for recall
  - **False Positive Rate** (**FPR**) $= \frac{\#\text{ of false positives}}{\#\text{ of false positives} + \#\text{ of true negatives}}$

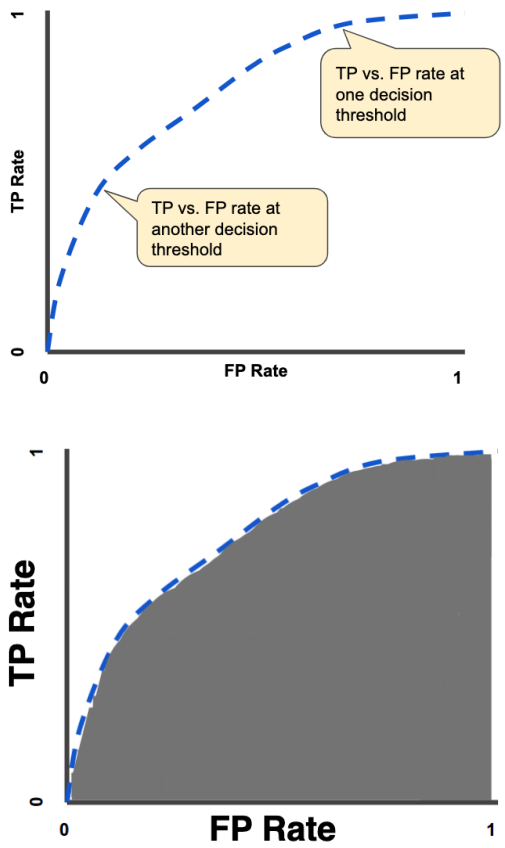


Figure 5. AUC (Area under the ROC Curve).

INTERPRETATION LOGISTIC REGRESSION RESULT (1)

- Pseudo $R^2 = 1 - \frac{\log(likelihood\ of\ data\ given\ the\ squiggle)}{\log(likelihood\ of\ data\ given\ overall\ \text{prob})}$
  - given $\#\ of\ 1 = 5; \#\ of\ 0 = 4; p = \frac{5}{9}$

    $\log(likelihood\ of\ data\ given\ overall\ prob) = 5 * \log\left(\frac{5}{9}\right) + 4 \times \log\left(\frac{4}{9}\right) = -2.69$
  - $R^2 = 1 - \frac{-1.64}{-2.69} = 0.39$

INTERPRETATION LOGISTIC REGRESSION RESULT (2)

- Coefficient:
  - Coefficient: 1 unit of X_i changed causes **how much** the log of the odds change
    - $\log\left(\frac{Y_1}{Y_0}\right) = a + b(X + \Delta X)$
  - np.exp of coefficient:1 unit of X_i changed causes **how many times** of the odds change
    - $\frac{\left(\frac{Y_1}{Y_0}\right)_{after}}{\left(\frac{Y_1}{Y_0}\right)_{before}} = \frac{e^{a+b(X+\Delta X)}}{e^{a+bX}} = e^{b\Delta X}$

```
logreg = LogisticRegression()
logreg.fit(X, y)
log_odds = logreg.coef_[0]

pd.DataFrame(log_odds,
             X.columns,
             columns=['coef'])\
            .sort_values(by='coef', ascending=False)
```

	coef
RM	1.833027
ZN	0.021019
TAX	-0.002191
INDUS	-0.047968
AGE	-0.054992
CRIM	-0.082490
DIS	-0.720439
NOX	-1.848019

Log odds

```
odds = np.exp(logreg.coef_[0])
pd.DataFrame(odds,
             X.columns,
             columns=['coef'])\
            .sort_values(by='coef', ascending=False)
```

	coef
RM	6.252784
ZN	1.021241
TAX	0.997812
INDUS	0.953164
AGE	0.946493
CRIM	0.920821
DIS	0.486538
NOX	0.157549

Odds

INTERPRETATION LOGISTIC REGRESSION RESULT (2)

- Classification Matrix
  - Overall accuracy
  - Class recall: high or low or moderate in each True class
  - Class precision: high or low or moderate in each Predicted class

accuracy: 68.48% +/- 0.93% (micro average: 68.48%)

	true false	true true	class precision
pred. false	2650	1092	70.82%
pred. true	301	377	55.60%
class recall	89.80%	25.66%	

HW4: APPLY LOGISTIC REGRESSION TO PREDICT **IMDB_SCORE** (LOW OR HIGH) ON IMDB

- movie_title : Title of the Movie
- duration: Duration in minutes
- director_name : Name of the Director of the Movie.
- director_facebook_likes : Number of likes of the Director on his Facebook Page.
- color: Film colorization. 'Black and White' or 'Color'
- genres: Film categorization like 'Animation', 'Comedy', 'Romance', 'Horror', 'Sci-Fi', 'Action', 'Family'
- actor_1_name: Primary actor starring in the movie
- actor_1_facebook_likes : Number of likes of the Actor_1 on his/her Facebook Page.
- actor_2_name: Other actor starring in the movie
- actor_2_facebook_likes : Number of likes of the Actor_2 on his/her Facebook Page.
- actor_3_name: Other actor starring in the movie
- actor_3_facebook_likes : Number of likes of the Actor_3 on his/her Facebook Page.
- num_critic_for_reviews : Number of critical reviews on imdb
- num_voted_users: Number of people who voted for the movie
- cast_total_facebook_likes: Total number of facebook likes of the entire cast of the movie.
- language : English, Arabic, Chinese, French, German, Danish, Italian, Japanese etc
- country: Country where the movie is produced.
- gross: Gross earnings of the movie in Dollars
- budget: Budget of the movie in Dollars
- title_year: The year in which the movie is released (1916:2016)
- imdb_score: IMDB Score of the movie on IMDB
- movie_facebook_likes: Number of Facebook likes in the movie page.

Drop out 10~20% in the middle

HW4: SOLUTION STEPS 1

- Load data: Merge & Concatenation
- Explore data:
  - Head()/tail()
  - info()→data_type & size
  - describe() & value_counts() plot → distribution
  - isna()→missing value

HW4: SOLUTION STEPS 2

- Preprocess data:
  - Split train-test
  - Encoding data features:
    - Integer → Float; Categorical → Binary Values
  - Fill in missing value in train dataset
  - Standardize features (comparable units) in train dataset
  - Select features in train dataset
    - Only numerical features
    - Correlation – use correlation matrix or heat map to ignore x which is highly correlated with another x

HW4: SOLUTION STEPS 3

- Model:
  - Debug the Logistic Regression model to make the model work for train dataset
  - Preprocess (based on train dataset) and predict test dataset
  - Evaluate classification result of test dataset
  - Interpret the Logistic Regression function