Logistic Regression Model

Peerapon S.

Department of Computer Engineering
King Mongkut's University of Technology Thonburi

Code and Data at `https://kmutt.me/cpe-ml`

Topics

- ☐ Model formulation and fitting
- ☐ Model interpretation
- ☐ Pseudo R^2 measures
- ☐ Test for significance of model
- ☐ Classification performance metrics

Classification Problem

	A	B	C	D	E
1	Age	Income	Jobsatisfaction	Desire	Enrolls
2	<=30	High	No	Fair	No
3	<=30	High	No	Excellent	No
4	31 to 40	High	No	Fair	Yes
5	>40	Medium	No	Fair	Yes
6	>40	Low	Yes	Fair	Yes
7	>40	Low	Yes	Excellent	No
8	31 to 40	Low	Yes	Excellent	Yes
9	<=30	Medium	No	Fair	No
10	<=30	Low	Yes	Fair	Yes
11	>40	Medium	Yes	Fair	Yes
12	<=30	Medium	Yes	Excellent	Yes
13	31 to 40	Medium	No	Excellent	Yes
14	31 to 40	High	Yes	Fair	Yes
15	>40	Medium	No	Excellent	No
16	<=30	Medium	Yes	Fair	

$$\boldsymbol{X} = [X_1, X_2, \ldots, X_p]$$

$$Y \in \{c_1, c_2, \ldots, c_K\}$$

For unseen $\boldsymbol{x}_0$, what is the most likely value of y_0 ?

Baseline: **Null classifier** that predicts class c_i using $\Pr\{Y = c_i\}$

Types of Categorical Responses

- Binary/Dichotomous response -- {0, 1}
- Binomial response
  - Discrete counts of "success" and "failure" from N trials
  - Repeated observations of binary response
- Polytomous (Multinomial) response -- {0, 1, 2, ..., C}
  - Nominal
  - Ordinal

Bernoulli Distribution

- Model a variable or response with two outcomes (binary response), each with probability p and $1-p$ respectively.
- Let $Y = 0$ if failure, $Y = 1$ if success.
- Y has a Bernoulli distribution with parameter p, or $Y \sim \text{Bern}(p)$ if

$$\mathbb{P}\{Y = 0\} = p$$

$$\mathbb{P}\{Y = 1\} = 1 - p$$

$$\mathbb{P}\{Y = i\} = p^i(1-p)^{1-i}, \quad i = 0, 1$$

$$\mathbb{E}\{Y\} = p$$

$$\mathbb{V}\text{ar}\{Y\} = p(1-p)$$

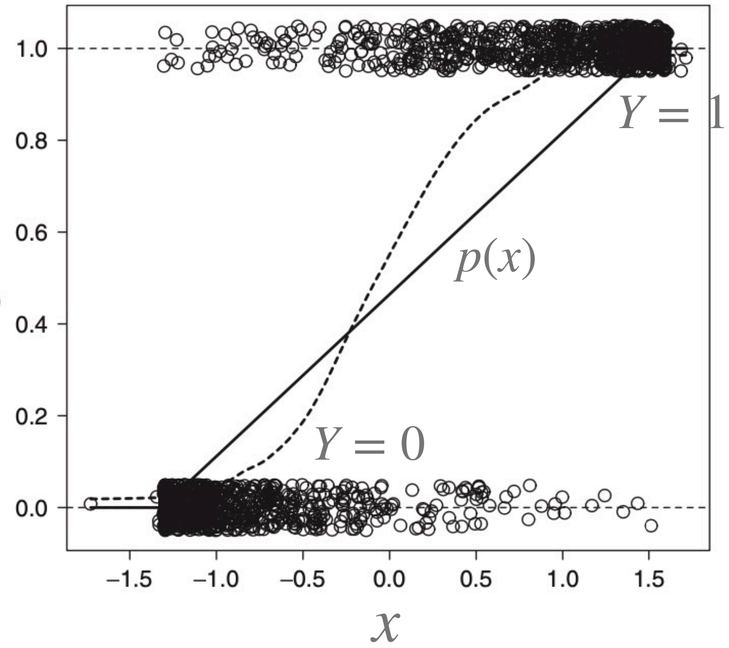
Linear Regression Model on Binary Data ?

- For a linear regression model (possibly with interaction terms),

$$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_m x_m + \epsilon$$

$$\mathbb{E}\{Y|\boldsymbol{x}\} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_m x_m = \boldsymbol{x}'\boldsymbol{\beta}$$

- Inappropriate if Y is a binary response, i.e., Bernoulli rvs with probability depending on $\boldsymbol{x}$.
  - Ex: Pass/Fail, Live/Death, Up/Down, etc.
  - E$\{Y|\boldsymbol{x}\} = p(\boldsymbol{x}) \in [0, 1]$ but $\boldsymbol{x}'\boldsymbol{\beta} \in \mathbf{R}$



Transformation of Predicted Probability

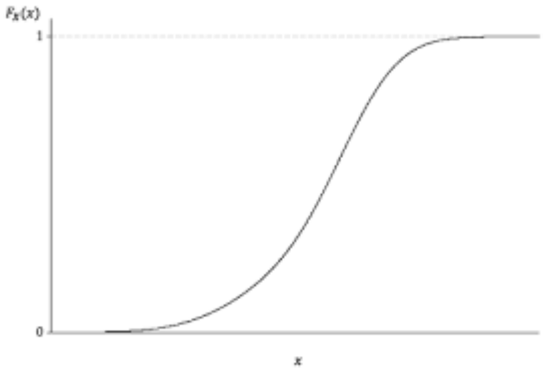
- For a binary response $Y \in \{0, 1\}$, define

$$p(\boldsymbol{x}) \triangleq \mathbb{P}\{Y = 1 \mid \boldsymbol{X} = \boldsymbol{x}\}, \quad \boldsymbol{x} = [x_1, x_2, \ldots, x_p]'$$

$$\mathbb{E}\{Y|\boldsymbol{x}\} = p(\boldsymbol{x})(1) + (1 - p(\boldsymbol{x}))(0) = p(\boldsymbol{x})$$

- To ensure $p(\boldsymbol{x}) \in [0,1]$ for a given $\boldsymbol{x}$, we can use any cumulative distribution function $F(\cdot)$ to transform a ***linear predictor*** $z = \boldsymbol{x}'\boldsymbol{\beta}$ into a unit interval:

$$\begin{aligned} p(\boldsymbol{x}) = \mathbb{E}\{Y|\boldsymbol{x}\} &= F(z) \\ &= F(\boldsymbol{x}'\boldsymbol{\beta}) \\ F^{-1}(p(\boldsymbol{x})) &= \boldsymbol{x}'\boldsymbol{\beta} \end{aligned}$$

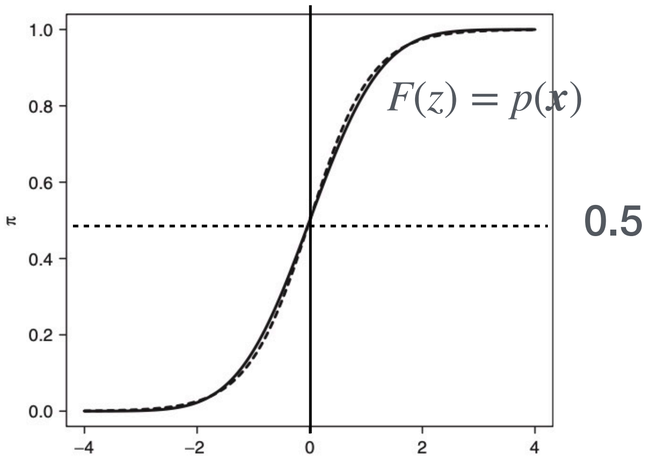


Logistic Regression Model

☐ $F(\cdot)$ chosen as the CDF of standard logistic distribution (sigmoid function):

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

$$\text{pdf} \quad f(z) = \frac{e^z}{[1+e^z]^2}, \quad \mu = 0,\ \sigma^2 = \pi^2/3$$



☐ Result in the ***logit*** or ***logistic regression*** model

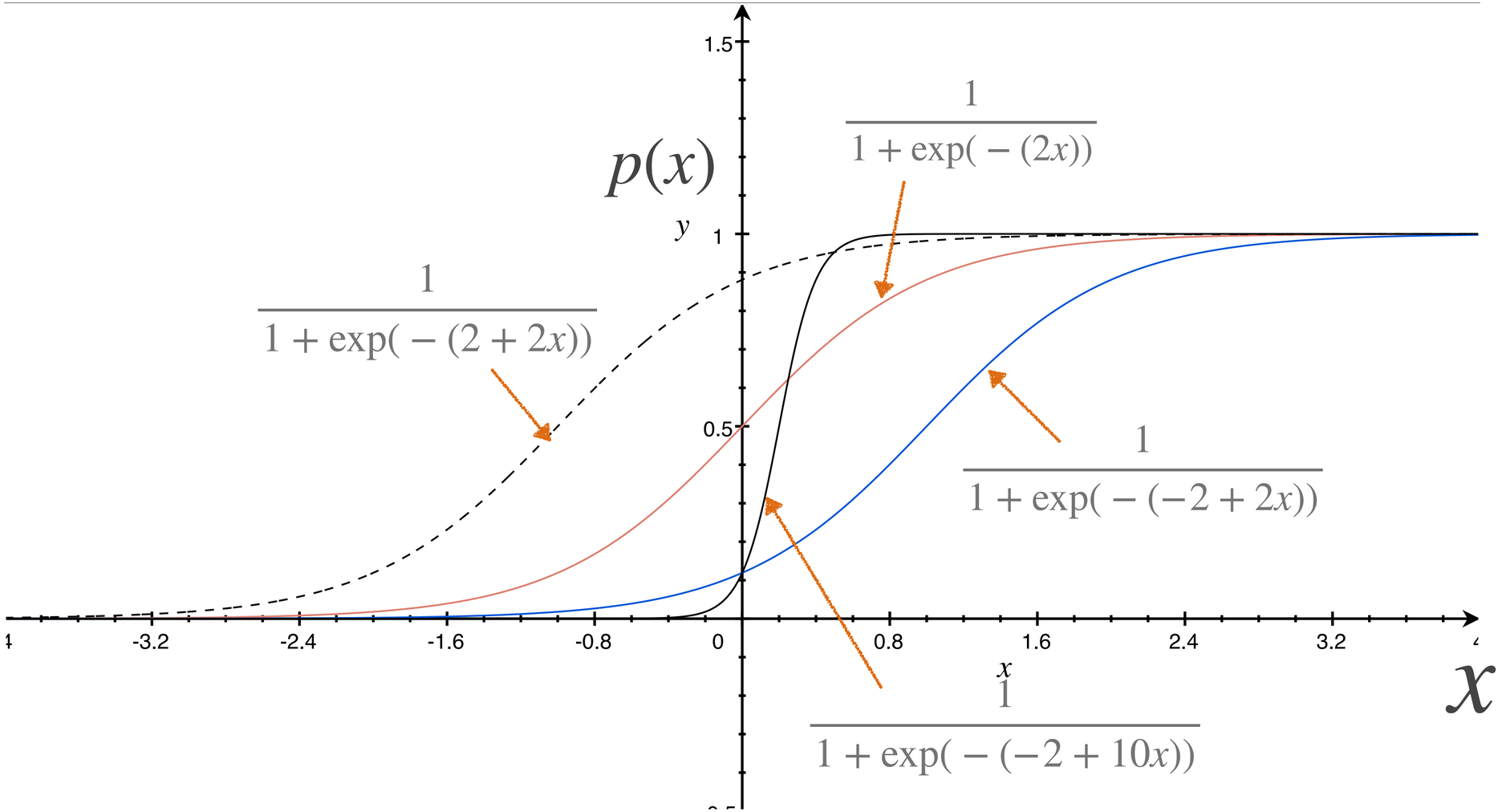
$$\mathbb{P}\{Y = 1 \mid \boldsymbol{x}\} = p(\boldsymbol{x}) = F(\boldsymbol{x}'\boldsymbol{\beta}) = \frac{1}{1+e^{-\boldsymbol{x}'\boldsymbol{\beta}}}$$

☐ The model estimates $p(\boldsymbol{x}) = P\{Y = 1 \,|\, \boldsymbol{x}\}$ that the response will be in one of the two classes.

☐ Classification rule to minimize the error rate (maximize the accuracy):

- Predict positive class $(Y = 1)$ if $p(\boldsymbol{x}) > 0.5$.
- Predict negative class $(Y = 0)$ if $p(\boldsymbol{x}) \leq 0.5$.

- Ex: For a single predictor, $p(x) = F(\beta_0 + \beta_1 x) = \dfrac{1}{1 + e^{-(\beta_0 + \beta_1 x)}}$
- How do the intercept and the coefficient affect the curve?



Log Odds

☐ Consider the odds of two classes:

$$\frac{p(\boldsymbol{x}_0)}{1-p(\boldsymbol{x}_0)} = e^{\boldsymbol{x}_0'\boldsymbol{\beta}}$$

$$\log \frac{p(\boldsymbol{x}_0)}{1-p(\boldsymbol{x}_0)} = \boldsymbol{x}_0'\boldsymbol{\beta}$$

- The log-odds $F^{-1}(p) = \log[p/(1-p)]$ is called the ***logit*** of p.
- Log of ***odds*** that Y is 1 rather than 0.

☐ The logit acts as a **linear discriminant function** (linear decision boundary). Why?

Model Fitting: Maximum Likelihood Estimation (MLE)

☐ Denote

- Response data as $y_1, y_2, \ldots, y_n$, and a given set of p predictors $x_1, x_2, \ldots, x_p$,
- $\boldsymbol{\beta} = [\beta_0, \beta_1, \ldots, \beta_p]$ as the unknown coefficients to be estimated.

☐ The likelihood function (chance of seeing the observed data) is the joint pmf

$$L(\boldsymbol{p}) = \mathbb{P}\{Y_1 = y_1, Y_2 = y_2, \ldots, Y_n = y_n | \boldsymbol{p}\}$$
$$= \prod_{i=1}^{n} \mathbb{P}\{Y_i = y_i | p_i\}$$
$$\mathbb{P}\{Y_i = y_i | p_i\} = p_i^{y_i}(1-p_i)^{1-y_i}, \quad Y_i \in \{0, 1\}$$
$$p_i = \frac{1}{1 + e^{-\boldsymbol{x}_i'\boldsymbol{\beta}}}$$

☐ Find $\boldsymbol{\beta} = [\beta_0, \beta_1, \ldots, \beta_m]$ that maximizes log $L(\boldsymbol{p})$. Needed later to assess the model quality-of-fit.

Loss Function

- To find $\boldsymbol{\beta}$, we form the log-likelihood function as the loss function and then determine its partial derivative.

$$\begin{aligned}
\log L(\boldsymbol{p}) = J(\boldsymbol{\beta}) &= \log \prod_{i=1}^{n} \mathbb{P}\{Y_i = y_i | p_i\} \\
&= \sum_{i=1}^{n} \log \mathbb{P}\{Y_i = y_i | p_i\} \\
&= \sum_{i=1}^{n} y_i \log p_i + (1 - y_i) \log(1 - p_i) \\
\frac{\partial J(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} &= \frac{1}{n} \sum_{i=1}^{n} (y_i - p_i(\boldsymbol{\beta})) \boldsymbol{x}_i
\end{aligned}$$

- The value of $\boldsymbol{\beta}$ that maximizes $J(\boldsymbol{\beta})$ can be determined by using the gradient ascent method or other optimization methods to solve for $\boldsymbol{\beta}$ at which $\partial J(\boldsymbol{\beta}) / \partial \boldsymbol{\beta} = 0$.

Regularization

☐ The penalty term $r(\boldsymbol{\beta})$ and the regularization strength C can be added to $J(\boldsymbol{\beta})$ to prevent overfitting as

$$J(\boldsymbol{\beta}) = \sum_{i=1}^{n} y_i \log p_i + (1 - y_i) \log(1 - p_i) + \frac{r(\boldsymbol{\beta})}{C}$$

$$\begin{aligned}
\text{None:} \quad & r(\boldsymbol{\beta}) = 0 \\
l_1: \quad & r(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_1 \\
l_2: \quad & r(\boldsymbol{\beta}) = \tfrac{1}{2}\|\boldsymbol{\beta}\|_2^2 = \tfrac{1}{2}\boldsymbol{\beta}^T\boldsymbol{\beta} \\
\text{Elastic net:} \quad & r(\boldsymbol{\beta}) = \tfrac{1-\rho}{2}\boldsymbol{\beta}^T\boldsymbol{\beta} + \rho\|\boldsymbol{\beta}\|_1
\end{aligned}$$

Example

- Detection ailing financial and business establishments is an important function of audit and control. Systematic failure to do audit and control can lead to grave consequences, such as the savings-and-loan fiasco of the 1980s in the United States. The dataset gives the operating financial ratios of 33 firms that went bankrupt after 2 years and 33 that remained solvent during the same period, where

$$X_1 = \frac{\text{Retained Earnings}}{\text{Total Assets}}$$

$$X_2 = \frac{\text{Earnings Before Interest and Taxes}}{\text{Total Assets}}$$

$$X_3 = \frac{\text{Sales}}{\text{Total Assets}}$$

$$Y = \begin{cases} 0, & \text{if bankrupt after 2 years} \\ 1, & \text{if solvent after 2 years} \end{cases}$$

Y	Label	X1	X2	X3
1	C1-Solvent	43	16.4	1.3
1	C1-Solvent	47	16	1.9
1	C1-Solvent	-3.3	4	2.7
1	C1-Solvent	35	20.8	1.9
1	C1-Solvent	46.7	12.6	0.9
1	C1-Solvent	20.8	12.5	2.4
1	C1-Solvent	33	23.6	1.5
1	C1-Solvent	26.1	10.4	2.1
1	C1-Solvent	68.6	13.8	1.6
1	C1-Solvent	37.3	33.4	3.5
1	C1-Solvent	59	23.1	5.5
1	C1-Solvent	49.6	23.8	1.9
1	C1-Solvent	12.5	7	1.8
1	C1-Solvent	37.3	34.1	1.5
1	C1-Solvent	35.3	4.2	0.9
1	C1-Solvent	49.5	25.1	2.6

Load dataset from sheet 'Financial' in data/logistic-regression.xlsx

```
import statsmodels.formula.api as smf
import statsmodels.api as sm

fin_df = pd.read_excel('data/logistic-regression.xlsx',
                       sheet_name='Financial', usecols=['Status','X1','X2','X3'], header=0)
```

```
fin_df.head()
fin_df.info()

fin_df['Status'].value_counts()
```

	Status	X1	X2	X3
0	C1-Solvent	43.0	16.4	1.3
1	C1-Solvent	47.0	16.0	1.9
2	C1-Solvent	-3.3	4.0	2.7
3	C1-Solvent	35.0	20.8	1.9
4	C1-Solvent	46.7	12.6	0.9

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 66 entries, 0 to 65
Data columns (total 4 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Status  66 non-null     object
 1   X1      66 non-null     float64
 2   X2      66 non-null     float64
 3   X3      66 non-null     float64
```

```
C1-Solvent     33
C2-Bankrupt    33
Name: Status, dtype: int64
```

Replace y values if they are labels.

```python
# Label encoding
fin_df['Y'] = fin_df['Status'].replace({"C1-Solvent":1, "C2-Bankrupt":0})

import statsmodels.formula.api as smf

# Use Formula style
formula = 'Y ~ X1+X2+X3'
solvent_model = smf.logit(formula = formula, data=fin_df).fit()

import statsmodels.api as sm

# Use API style
y_fin = fin_df['Y']
X_fin = sm.add_constant(fin_df.drop(['Y', 'Status'], axis=1))
solvent_model = sm.Logit(y_fin, X_fin).fit()

solvent_model.params
```

Y needs to be numeric !

```
Optimization terminated successfully.
         Current function value: 0.044037
         Iterations 14
[21]:
const   -10.153
X1        0.331
X2        0.181
X3        5.087
dtype: float64
```

$$\hat{p}(\boldsymbol{x}) = \frac{1}{1+\exp(-(-10.153+0.331x_1+0.181x_2+5.087x_3)}$$

$$\hat{\beta}_0 = -10.153$$

$$\hat{\beta}_1 = 0.331$$

$$\hat{\beta}_2 = 0.181$$

$$\hat{\beta}_3 = 5.087$$

```
# Predict the probabilities of being solvent

predictions = solvent_model.predict(fin_df[20:22]) # Model fitted with either styles

predictions = solvent_model.predict(np.array([[1, 31, 4.2, 0.3],
                                              [1, 19.2, 1.3, 1.1]])) # Only API-style fitted model
```

Deviance

- L_F : Maximized likelihood of the full model
- L_0 : Maximized likelihood of the null model, e.g., constant P(Class 1)
- $G_F^2 = -2 \log L_F$: Residual deviance of the (full) model
- $G_0^2 = -2 \log L_0$: Null deviance

Log-likelihood of full model $\log(L_F)$: `solvent_model.llf`
Log-likelihood of null model $\log(L_0)$: `solvent_model.llnull`

Pseudo R^2 Measures

☐ Quality-of-fit analogous to a linear model can be measured by various quantities:

$$\text{McFadden's } R^2 = 1 - \frac{G_F^2}{G_0^2} = 1 - \frac{\log L_F}{\log L_0}$$

$$\text{Adjusted McFadden's } R^2 = 1 - \frac{\log L_F - m}{\log L_0}$$

$$\text{Cox \& Snell } R^2 = 1 - \left(\frac{L_0}{L_F}\right)^{2/N}$$

$$\text{Nagelkerke-Cragg \& Uhler's } R^2 = \frac{R^2_{\text{Cox\&Snell}}}{1 - L_0^{2/N}}$$

m = # model parameters - 1
N = # observations

Log-likelihood of full model $\log(L_F)$: `solvent_model.llf`
Log-likelihood of null model $\log(L_0)$: `solvent_model.llnull`
Number of model parameters - 1 m: `solvent_model.df_model`

```
McFadden's R2: 0.936
Adjusted McFadden's R2: 0.871
Cox & Snell R2: 0.727
Nagelkerke R2: 0.969
```

Likelihood Ratio Test for Significance of Model

- Does the model do better than predicting by just using the constant probability?
- For m regressors/predictors:

$$H_0 : \beta_i = 0, \forall i = 1, 2, \ldots, m$$

$$H_a : \text{At least one coefficient (not } \beta_0) \text{ is not zero}$$

- Under large sample, the likelihood ratio test statistic (G) under H_0 is

$$G = G_0^2 - G_F^2 \sim \chi^2(m)$$

$$G = 2[\log(L_F) - \log(L_0)] = 2 \log \frac{L_F}{L_0} \sim \chi^2(m)$$

- From the solvent data,

$$G = 85.683$$

$$\chi^2_{3;0.05} = 7.814$$

```
print(solvent_model.summary2())
```

```
 Results: Logit
=================================================================
Model:              Logit            Pseudo R-squared: 0.936
Dependent Variable: Y                AIC:              13.8129
Date:               2024-08-01 13:41 BIC:              22.5715
No. Observations:   66               Log-Likelihood:   -2.9065
Df Model:           3                LL-Null:          -45.748
Df Residuals:       62               LLR p-value:      1.8520e-18
Converged:          1.0000           Scale:            1.0000
No. Iterations:     14.0000
-----------------------------------------------------------------
         Coef.     Std.Err.      z      P>|z|     [0.025     0.975]
-----------------------------------------------------------------
const   -10.1535    10.8401   -0.9367   0.3489   -31.3996   11.0927
X1        0.3312     0.3007    1.1014   0.2707    -0.2582    0.9207
X2        0.1809     0.1069    1.6916   0.0907    -0.0287    0.3904
X3        5.0875     5.0821    1.0010   0.3168    -4.8733   15.0483
=================================================================
```

McFadden's R^2

Try refitting without intercept.

Classification Model Evaluation -- Performance Metrics

- Use prediction errors to compute the model performance
  - Train error - Prediction error evaluated on the train data.
  - Test error - Prediction error evaluated on the test data.
- For a regression problem, mean squared errors (MSE) is computed directly from prediction errors.
- Basic classification performance/scoring metrics/measures
  - Accuracy / Misclassification rate
  - Precision
  - Recall
  - F1 score

Confusion Matrix

TP: True Positive
TN: True Negative
FP: False Positive
FN: False Negative

Actual class	Predicted Class		
	Negative	Positive	
Negative	250 (TN)	50 (FP)	300
Positive	150 (FN)	550 (TP)	700
	400	600	

Accuracy (Total Error Rate)

TP: True Positive
TN: True Negative
FP: False Positive
FN: False Negative

Actual class	Predicted Class	
	Negative	Positive
Negative	250 (TN)	50 (FP)
Positive	150 (FN)	550 (TP)
	400	600

300
700

$$\text{Accuracy} = \frac{TP + TN}{N} = \frac{550 + 250}{1000} = 0.8$$

Precision and Recall

TP: True Positive
TN: True Negative
FP: False Positive
FN: False Negative

Actual class	Predicted Class: Negative	Predicted Class: Positive	
Negative	250 (TN)	50 (FP)	300
Positive	150 (FN)	550 (TP)	700
	400	600	

Positive Class

$$\text{Precision} = \frac{TP}{TP + FP} = \frac{550}{600} = 0.916$$

%correct+ among those predicted+ (TPR)

$$\text{Recall} = \frac{TP}{TP + FN} = \frac{550}{700} = 0.785$$

%correct+ among those actual+

Negative Class

$$\text{Precision} = \frac{TN}{TN + FN} = \frac{250}{400} = 0.625$$

$$\text{Recall} = \frac{TN}{TN + FP} = \frac{250}{300} = 0.833$$

F1-score

TP: True Positive
TN: True Negative
FP: False Positive
FN: False Negative

Actual class	Predicted Class	
	Negative	Positive
Negative	250 (TN)	50 (FP)
Positive	150 (FN)	550 (TP)

	400	600

300
700

Positive Class

$$\mathrm{F1} = \frac{2}{1/\mathrm{Precision} + 1/\mathrm{Recall}} = \frac{2TP}{2TP + FP + FN} = 0.845$$

```python
# Generate classification report
from sklearn.metrics import confusion_matrix, classification_report

predictions = solvent_model.predict()
predictions_nominal = [ 0 if x < 0.5 else 1 for x in predictions]
predictions_nominal = (solvent_model.predict() > 0.5).astype("int32")

print(confusion_matrix(fin_df["Y"], predictions_nominal))
print(classification_report(fin_df["Y"], predictions_nominal, digits = 3))
```

```
[[32  1]
 [ 1 32]]
              precision    recall  f1-score   support

           0      0.970     0.970     0.970        33
           1      0.970     0.970     0.970        33

    accuracy                          0.970        66
   macro avg      0.970     0.970     0.970        66
weighted avg      0.970     0.970     0.970        66
```

Average of each metric weighted by the number of support

Average of each metric

Sklearn Logistic Regression

```
# Load data from file to fin_df

fin_df = pd.read_excel('data/logistic-regression.xlsx',
                       sheet_name='Financial', usecols=['Status','X1','X2','X3'], header=0)
y = fin_df['Status']
X = fin_df.drop('Status', axis=1)
```

Y can be categorical !

```
from sklearn.linear_model import LogisticRegression

logreg_clf = LogisticRegression(fit_intercept=True, penalty=None)
logreg_clf.fit(X, y)
```

class `sklearn.linear_model.`**`LogisticRegression`**(*penalty='l2', *, dual=False, tol=0.0001, C=1.0, fit_intercept=True, intercept_scaling=1, class_weight=None, random_state=None, solver='lbfgs', max_iter=100, multi_class='auto', verbose=0, warm_start=False, n_jobs=None, l1_ratio=None*) [source]

```
# Predict labels and probabilities
logreg_clf.predict(X[:10])
logreg_clf.predict_proba(X[:10]).round(3)
```

```
array(['C2-Bankrupt', 'C1-Solvent', 'C1-Solvent', 'C2-Bankrupt',
       'C2-Bankrupt', 'C1-Solvent', 'C2-Bankrupt', 'C2-Bankrupt',
       'C2-Bankrupt', 'C1-Solvent'], dtype=object)
array([[0., 1.],
       [1., 0.],
       [1., 0.],
       [0., 1.],
       [0., 1.],
       [1., 0.],
       [0., 1.],
       [0., 1.],
       [0., 1.],
       [1., 0.]])
```

```python
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report

y_pred = logreg_clf.predict(X)

y_pred_accuracy = logreg_clf.score(X, y)
print(f'Accuracy of logistic regression classifier on train set: {y_pred_accuracy:.2f}')

print(confusion_matrix(y, y_pred))
print(classification_report(y, y_pred))
```

```
[[32  1]
 [ 1 32]]
              precision    recall  f1-score   support

  C1-Solvent       0.97      0.97      0.97        33
 C2-Bankrupt       0.97      0.97      0.97        33

    accuracy                           0.97        66
   macro avg       0.97      0.97      0.97        66
weighted avg       0.97      0.97      0.97        66
```

Summary

- Logistic regression applies transformation to a linear predictor
  - Fitted linear predictor interpreted as log odds ratio
  - Coefficient related to change in odds ratio for a unit increase of variable
- Model coefficients solved from MLE method
  - Likelihood function is formed.
  - Solved for coefficients maximizing the joint occurrence of data.
- Likelihood ratio for testing significance of model and coefficients and deriving pseudo R^2.

In-Class Activity - Default Model using Logistic Regression

- From the credit default data set, build a logistic regression model that predicts the default rate as a function of
  - Student status (Model A)
  - Student status and Balance (Model B)
  - Student status, Balance, and Income (Model C)
- From the models above, interpret the model results regarding the effect of the predictors on the response.
- Evaluate the model performance on the train data for the model containing highest number of significant variables.

Data from sheet 'Default' in logistic-regression.xlsx

No.	default	student	balance	income
1	No	No	729.526495	44361.6251
2	No	Yes	817.180407	12106.1347
3	No	No	1073.54916	31767.1389
4	No	No	529.250605	35704.4939
5	No	No	785.655883	38463.4959
6	No	Yes	919.58853	7491.55857
7	No	No	825.513331	24905.2266
8	No	Yes	808.667504	17600.4513
9	No	No	1161.05785	37468.5293
10	No	No	0	29275.2683
11	No	Yes	0	21871.0731
12	No	Yes	1220.58375	13268.5622
13	No	No	237.045114	28251.6953
14	No	No	606.742343	44994.5558
15	No	No	1112.9684	23810.1741
16	No	No	286.23256	45042.413
17	No	No	0	50265.3124
18	No	Yes	527.540184	17636.5396
19	No	No	485.936864	61566.1061