Dative_Alternation_Analysis_Using_Logistic_Regression

June 2, 2023

0.1 A new application of logistic regression: the dative alternation

The work you need to do for this pset involves applying logistic regression to a new case, the **dative alternation**, which we studied in a previous pset. We will use the `dative` dataset from Bresnan et al. (2007). First we load the dataset:

[ ]:
```
# imports
import statsmodels.api as sm
import math, random
import pandas as pd
import numpy as np
```

[ ]:
```
dat = pd.read_csv("https://gist.githubusercontent.com/scaperex/
 ↪278815a736401d36021aa9fe31b9a0cb/raw/
 ↪cf338a8cf745fa5820c4ea97af682d265bc1a34f/dative-alternation.csv")
dat
```

[ ]:
```
      Unnamed: 0 Speaker Modality   Verb SemanticClass  LengthOfRecipient  \
0              1     NaN  written   feed             t                  1
1              2     NaN  written   give             a                  2
2              3     NaN  written   give             a                  1
3              4     NaN  written   give             a                  1
4              5     NaN  written  offer             c                  2
…            …     …      …     …           …                …
3258        3258   S1190   spoken   tell             c                  1
3259        3259   S1423   spoken   give             a                  1
3260        3260   S1680   spoken   give             a                  4
3261        3261   S1680   spoken   give             a                  1
3262        3262   S1023   spoken    pay             a                  1

     AnimacyOfRec  DefinOfRec    PronomOfRec  LengthOfTheme AnimacyOfTheme  \
0         animate    definite     pronominal             14      inanimate
1         animate    definite  nonpronominal              3      inanimate
2         animate    definite  nonpronominal             13      inanimate
3         animate    definite     pronominal              5      inanimate
4         animate    definite  nonpronominal              3      inanimate
…             …         …            …              …          …
3258      animate    definite     pronominal              1      inanimate
```

```
3259       animate    definite     pronominal              9      inanimate
3260       animate  indefinite  nonpronominal              2      inanimate
3261     inanimate    definite     pronominal              2      inanimate
3262       animate    definite     pronominal              2      inanimate

     DefinOfTheme  PronomOfTheme RealizationOfRecipient AccessOfRec  \
0      indefinite  nonpronominal                     NP       given
1      indefinite  nonpronominal                     NP       given
2        definite  nonpronominal                     NP       given
3      indefinite  nonpronominal                     NP       given
4        definite  nonpronominal                     NP       given
…             …              …                    …         …
3258     definite     pronominal                     NP       given
3259   indefinite  nonpronominal                     NP       given
3260     definite  nonpronominal                     PP  accessible
3261   indefinite  nonpronominal                     NP       given
3262   indefinite  nonpronominal                     NP       given

     AccessOfTheme
0              new
1              new
2              new
3              new
4              new
…              …
3258         given
3259    accessible
3260    accessible
3261    accessible
3262    accessible

[3263 rows x 16 columns]
```

We see that it uses text values for some of the variables we are interested in (the response variable `RealizationOfRecipient`, and the variables expressing length and pronominality of theme and object). We create numeric versions of these variables, arbitrarily coding a double object outcome as `1` ("success") and a prepositional dative outcome as `0`.

[ ]:
```
dat["Response"] = [1 if x =="NP" else 0 for x in dat["RealizationOfRecipient"]]
dat["RecPro"] = [1 if x =="pronominal" else 0 for x in dat["PronomOfRec"]]
dat["ThemePro"] = [1 if x =="pronominal" else 0 for x in dat["PronomOfTheme"]]
dat[["RealizationOfRecipient","Response","PronomOfRec","RecPro","ThemePro"]]
```

[ ]:
```
     RealizationOfRecipient  Response    PronomOfRec  RecPro  ThemePro
0                        NP         1     pronominal       1         0
1                        NP         1  nonpronominal       0         0
2                        NP         1  nonpronominal       0         0
```

```
3                       NP         1     pronominal       1          0
4                       NP         1  nonpronominal       0          0
…                     …       …             …      …       …
3258                    NP         1     pronominal       1          1
3259                    NP         1     pronominal       1          0
3260                    PP         0  nonpronominal       0          0
3261                    NP         1     pronominal       1          0
3262                    NP         1     pronominal       1          0

[3263 rows x 5 columns]
```

[ ]:

```
## TODO: create numeric variables for PronomOfTheme
dat["ThemePro"] = [1 if x =="pronominal" else 0 for x in dat["PronomOfTheme"]]
dat[["ThemePro"]]
```

[ ]:

```
      ThemePro
0            0
1            0
2            0
3            0
4            0
…         …
3258         1
3259         0
3260         0
3261         0
3262         0

[3263 rows x 1 columns]
```

To capture the possibility of an overall preference for one construction or the other, we add an "intercept" term to the logistic regression model, by creating a new `Dummy` variable in the data frame. We then fit a baseline model using only the intercept and find that there is an overall majority preference for the **DO** realization in this dataset (the intercept's fitted weight is greater than 0). We also see that the intercept-only model simply recapitulates the sample mean.

[ ]:

```
dat["Dummy"] = 1
x = dat[["Dummy"]]
y = dat[["Response"]]
model_0 = sm.GLM(y,x,family=sm.families.Binomial()) # first argument is␣
↪response, second argument is predictor matrix, third argument says this is␣
↪logistic regression
model_0 = model_0.fit()
print(model_0.summary())
print("Predicted proportion of DO outcomes based on fitted intercept-only model:
↪", round(np.mean(model_0.predict(x)),4))
```

```
print("Proportion of data with DO outcome:", round(np.mean(y["Response"]),4)) #␣
↪same as model-predicted proportion
```

```
                 Generalized Linear Model Regression Results
==============================================================================
Dep. Variable:               Response   No. Observations:                 3263
Model:                            GLM   Df Residuals:                     3262
Model Family:                Binomial   Df Model:                            0
Link Function:                  Logit   Scale:                          1.0000
Method:                          IRLS   Log-Likelihood:                -1870.5
Date:                Sat, 13 May 2023   Deviance:                       3741.1
Time:                        19:21:49   Pearson chi2:                 3.26e+03
No. Iterations:                     4   Pseudo R-squ. (CS):         -2.220e-16
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Dummy          1.0450      0.040     26.189      0.000       0.967       1.123
==============================================================================
Predicted proportion of DO outcomes based on fitted intercept-only model: 0.7398
Proportion of data with DO outcome: 0.7398
```

Task: In the below code boxes, complete the five parts of the problem specified in the pset PDF.

[ ]:
```
def calc_accuracy(predicted_values, true_values):
  s = 0

  for pred_val, true_val in zip(predicted_values, true_values):

    if pred_val > 0.5:
      pred_val = 1

    else:
      pred_val = 0

    if pred_val == true_val:
      s += 1

  acc = s / len(true_values)
  return acc


def calc_loglikelihood(predicted_values, true_values):
  log_likelihood = 0

  for pred_val, true_val in zip(predicted_values, true_values):
    # we use log with base 2 according to the course's convention
```

```
    log_likelihood += (true_val * math.log2(pred_val)) + ((1 - true_val) * math.
 ↪log2(1 - pred_val))

  return log_likelihood


def eval_model(model, test_set, predictors):
  X_test = test_set[predictors]
  y_predicted, y = model.predict(X_test), test_set["Response"]

  model_accuracy = calc_accuracy(y_predicted, y)
  model_log_likelihood = calc_loglikelihood(y_predicted, y)

  print(f"model accuracy on the test set: {round(model_accuracy, 3)}")
  print(f"model log-likelihood on the test set: {round(model_log_likelihood,␣
 ↪3)}")
```

0.1.1 Part 1

implementation of 80/20 train/test random split of the "dative" dataset

[ ]:
```
# setting "random_set" in order get consistent results
train_set = dat.sample(frac=0.8, random_state=0)
test_set = dat.drop(train_set.index.tolist())

# reset indicies
train_set.reset_index(drop=True, inplace=True)
test_set.reset_index(drop=True, inplace=True)

display(train_set)
display(test_set)
```

```
      Unnamed: 0 Speaker Modality    Verb SemanticClass  LengthOfRecipient  \
0           2464   S1421   spoken    give             t                  1
1           1515   S1151   spoken    send             t                  1
2           2756   S1462   spoken   teach             c                  1
3            655     NaN  written    give             a                  8
4           1998   S1259   spoken    tell             c                  1
…             …       …       …       …             …                  …
2605        2709   S1488   spoken    tell             c                  1
2606        1181   S1022   spoken    give             c                  1
2607         338     NaN  written     pay             t                  1
2608          47     NaN  written   offer             t                  1
2609         748     NaN  written  funnel             t                  4

     AnimacyOfRec  DefinOfRec    PronomOfRec  LengthOfTheme AnimacyOfTheme  \
0         animate    definite     pronominal              2       inanimate
```

```
1         animate    definite     pronominal              2      inanimate
2         animate    definite     pronominal              2      inanimate
3       inanimate  indefinite  nonpronominal              7      inanimate
4         animate    definite     pronominal              3      inanimate
…           …          …              …           …            …
2605      animate    definite     pronominal              3      inanimate
2606      animate    definite     pronominal              3      inanimate
2607      animate  indefinite  nonpronominal              6      inanimate
2608      animate  indefinite  nonpronominal              3      inanimate
2609      animate    definite  nonpronominal              1      inanimate

     DefinOfTheme  PronomOfTheme RealizationOfRecipient AccessOfRec  \
0      indefinite  nonpronominal                     NP       given
1      indefinite     pronominal                     NP       given
2      indefinite  nonpronominal                     NP       given
3      indefinite  nonpronominal                     PP         new
4        definite  nonpronominal                     NP       given
…           …             …                      …          …
2605     definite     pronominal                     NP       given
2606     definite  nonpronominal                     NP       given
2607   indefinite  nonpronominal                     NP  accessible
2608   indefinite  nonpronominal                     NP       given
2609   indefinite  nonpronominal                     PP         new

     AccessOfTheme  Response  RecPro  ThemePro  Dummy
0              new         1       1         0      1
1       accessible         1       1         1      1
2              new         1       1         0      1
3              new         0       0         0      1
4            given         1       1         0      1
…            …        …      …       …     …
2605         given         1       1         1      1
2606    accessible         1       1         0      1
2607           new         1       0         0      1
2608           new         1       0         0      1
2609           new         0       0         0      1

[2610 rows x 20 columns]

     Unnamed: 0 Speaker Modality   Verb SemanticClass  LengthOfRecipient  \
0             1     NaN  written   feed             t                  1
1             4     NaN  written   give             a                  1
2             8     NaN  written  bring             a                  1
3            22     NaN  written   give             a                  1
4            25     NaN  written   give             a                  2
..          …      …      …    …            …                …
648        3240   S1697   spoken   give             c                  1
649        3243   S1621   spoken   give             a                  1
```

```
650        3244   S1621   spoken    pay           a                 4
651        3251   S1382   spoken   tell           c                 1
652        3257   S1701   spoken   send           t                 1

    AnimacyOfRec  DefinOfRec    PronomOfRec  LengthOfTheme AnimacyOfTheme  \
0        animate    definite     pronominal             14      inanimate
1        animate    definite     pronominal              5      inanimate
2        animate    definite     pronominal              1      inanimate
3        animate    definite  nonpronominal              7      inanimate
4        animate  indefinite  nonpronominal             10      inanimate
..           …          …             …            …            …
648      animate    definite     pronominal              3      inanimate
649      animate    definite     pronominal              3      inanimate
650    inanimate    definite  nonpronominal             10      inanimate
651      animate    definite     pronominal              2      inanimate
652      animate    definite  nonpronominal              4      inanimate

    DefinOfTheme  PronomOfTheme RealizationOfRecipient AccessOfRec  \
0     indefinite  nonpronominal                     NP       given
1     indefinite  nonpronominal                     NP       given
2     indefinite  nonpronominal                     NP       given
3     indefinite  nonpronominal                     NP       given
4       definite  nonpronominal                     NP  accessible
..           …             …                    …          …
648   indefinite  nonpronominal                     NP       given
649   indefinite  nonpronominal                     NP       given
650   indefinite  nonpronominal                     PP  accessible
651   indefinite  nonpronominal                     NP       given
652   indefinite  nonpronominal                     PP         new

    AccessOfTheme  Response  RecPro  ThemePro  Dummy
0             new         1       1         0      1
1             new         1       1         0      1
2             new         1       1         0      1
3      accessible         1       0         0      1
4      accessible         1       0         0      1
..            …        …      …         …     …
648    accessible         1       1         0      1
649    accessible         1       1         0      1
650    accessible         0       0         0      1
651    accessible         1       1         0      1
652    accessible         0       0         0      1

[653 rows x 20 columns]
```

0.1.2 Part 2

Logistic regression model to the training set.

Predictors: recipient pronominality, intercept.

[ ]:
```
X = train_set[["RecPro", "Dummy"]]
y = train_set[["Response"]]

model_1 = sm.GLM(y, X, family=sm.families.Binomial())
model_1 = model_1.fit()
print(model_1.summary())
```

```
                 Generalized Linear Model Regression Results
==============================================================================
Dep. Variable:               Response   No. Observations:                 2610
Model:                            GLM   Df Residuals:                     2608
Model Family:                Binomial   Df Model:                            1
Link Function:                  Logit   Scale:                          1.0000
Method:                          IRLS   Log-Likelihood:                -1246.3
Date:                Sat, 13 May 2023   Deviance:                       2492.7
Time:                        19:21:49   Pearson chi2:                 2.61e+03
No. Iterations:                     5   Pseudo R-squ. (CS):             0.1668
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
RecPro         2.0750      0.102     20.406      0.000       1.876       2.274
Dummy          0.0142      0.064      0.223      0.824      -0.111       0.139
==============================================================================
```

[ ]:
```
eval_model(model_1, test_set, predictors=["RecPro", "Dummy"])
```

```
model accuracy on the test set: 0.723
model log-likelihood on the test set: -436.818
```

First, we can see that the model's accuracy on the test set is greater than 0.5. Thus, this model has some degree of predictive power, since it's better than a random classifier.

Second, it appears that $\beta_{RecPro} = 2.075$. Hence we can conlcude the following:

- Since $\beta_{RecPro} > 0$, the probability of the dative alternation being DO is higher when the recipient (in the sentence) is a pronoun

  $(x_{RecPro} = 1)$.
- The odds-ratio is $e^{2.075}$ times larger, when the recipient (in the sentence) is a pronoun $(x_{RecPro} = 1)$.

0.1.3 Part 3

Logistic regression model to the training set.

Predictors: recipient pronominality, theme pronominality, intercept.

```
[ ]: X = train_set[["RecPro", "ThemePro", "Dummy"]]
     y = train_set[["Response"]]

     model_2 = sm.GLM(y, X, family=sm.families.Binomial())
     model_2 = model_2.fit()
     print(model_2.summary())
```

```
                 Generalized Linear Model Regression Results
==============================================================================
Dep. Variable:               Response   No. Observations:                 2610
Model:                            GLM   Df Residuals:                     2607
Model Family:                Binomial   Df Model:                            2
Link Function:                  Logit   Scale:                          1.0000
Method:                          IRLS   Log-Likelihood:                -1046.7
Date:                Sat, 13 May 2023   Deviance:                       2093.4
Time:                        19:21:49   Pearson chi2:                 2.58e+03
No. Iterations:                     6   Pseudo R-squ. (CS):             0.2850
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
RecPro         2.8476      0.140     20.371      0.000       2.574       3.122
ThemePro      -3.0000      0.168    -17.893      0.000      -3.329      -2.671
Dummy          0.2297      0.067      3.421      0.001       0.098       0.361
==============================================================================
```

```
[ ]: eval_model(model_2, test_set, predictors=["RecPro", "ThemePro", "Dummy"])
```

```
model accuracy on the test set: 0.75
model log-likelihood on the test set: -379.969
```

First, we can see that both the model's accuracy and log-likelihood on the test set are greater than those of the previous one. Thus, we can conclude that this model has a greater predictive power (by these metrics).

Second, it appears that $\beta_{RecPro} = 2.848$. Hence we can conlcude the following:

- Since $\beta_{RecPro} > 0$, the probability of the dative alternation being DO is higher when the recipient (in the sentence) is a pronoun

  $(x_{RecPro} = 1)$.
- The odds-ratio is $e^{2.848}$ times larger, when the recipient (in the sentence) is a pronoun $(x_{RecPro} = 1)$.

Third, it appears that $\beta_{ThemePro} = -3.0$. Hence we can conlcude the following:

- Since $\beta_{ThemePro} < 0$, the probability of the dative alternation being DO is lower when the theme (in the sentence) is a pronoun

  $(x_{ThemePro} = 1)$.

- The odds-ratio is e^{-3} times smaller, when the recipient (in the sentence) is a pronoun ($x_{ThemePro} = 1$).

Moreover, we observe that $|\beta_{ThemePro}| > |\beta_{RecPro}|$ and $\beta_{ThemePro} < 0$. As a result, the probability of a sentence with both recipient and theme as pronouns being classified as DO, is lower than that of a sentence with none as pronouns (according to the model).

0.1.4 Part 4

In this part, we will test two more logostic regression models.

Part 4.1 Logistic regression model to the training set.

Predictors: recipient pronominality, theme pronominality, recipient length, theme length, intercept.

[ ]:
```
X = train_set[["RecPro", "ThemePro", "LengthOfRecipient", "LengthOfTheme",␣
 ↪"Dummy"]]
y = train_set[["Response"]]

model_3 = sm.GLM(y, X, family=sm.families.Binomial())
model_3 = model_3.fit()
print(model_3.summary())
```

```
                 Generalized Linear Model Regression Results
==============================================================================
Dep. Variable:               Response   No. Observations:                 2610
Model:                            GLM   Df Residuals:                     2605
Model Family:                Binomial   Df Model:                            4
Link Function:                  Logit   Scale:                          1.0000
Method:                          IRLS   Log-Likelihood:                -896.22
Date:                Sat, 13 May 2023   Deviance:                       1792.4
Time:                        19:21:49   Pearson chi2:                 3.45e+03
No. Iterations:                     6   Pseudo R-squ. (CS):             0.3629
Covariance Type:            nonrobust
================================================================================
=====
                        coef    std err          z      P>|z|      [0.025
0.975]
--------------------------------------------------------------------------------
-----
RecPro                2.4551      0.160     15.299      0.000       2.141
2.770
ThemePro             -2.7656      0.172    -16.103      0.000      -3.102
-2.429
LengthOfRecipient    -0.4065      0.042     -9.622      0.000      -0.489
-0.324
LengthOfTheme         0.2333      0.023     10.243      0.000       0.189
0.278
Dummy                 0.3342      0.152      2.193      0.028       0.036
```

```
0.633
==============================================================================
=====
```

```
[ ]: eval_model(model_3, test_set, predictors=["RecPro", "ThemePro",␣
      ↪"LengthOfRecipient", "LengthOfTheme", "Dummy"])
```

```
model accuracy on the test set: 0.864
model log-likelihood on the test set: -315.059
```

Part 4.2 Logistic regression model to the training set.

Predictors: recipient pronominality, theme pronominality, log recipient length, log theme length, intercept.

```
[ ]: # we use log with base 2 according to the course's convention
     train_set["LogLengthOfRecipient"] = [math.log2(x) for x in␣
      ↪train_set["LengthOfRecipient"]]
     train_set["LogLengthOfTheme"] = [math.log2(x) for x in␣
      ↪train_set["LengthOfTheme"]]

     test_set["LogLengthOfRecipient"] = [math.log2(x) for x in␣
      ↪test_set["LengthOfRecipient"]]
     test_set["LogLengthOfTheme"] = [math.log2(x) for x in test_set["LengthOfTheme"]]

     X = train_set[["RecPro", "ThemePro", "LogLengthOfRecipient",␣
      ↪"LogLengthOfTheme", "Dummy"]]
     y = train_set[["Response"]]

     model_4 = sm.GLM(y, X, family=sm.families.Binomial())
     model_4 = model_4.fit()
     print(model_4.summary())
```

```
                 Generalized Linear Model Regression Results
==============================================================================
Dep. Variable:               Response   No. Observations:                 2610
Model:                            GLM   Df Residuals:                     2605
Model Family:                Binomial   Df Model:                            4
Link Function:                  Logit   Scale:                          1.0000
Method:                          IRLS   Log-Likelihood:                -893.29
Date:                Sat, 13 May 2023   Deviance:                       1786.6
Time:                        19:21:49   Pearson chi2:                 2.74e+03
No. Iterations:                     6   Pseudo R-squ. (CS):             0.3643
Covariance Type:            nonrobust
==============================================================================
========
                           coef    std err          z      P>|z|      [0.025
0.975]
```

```
------------------------------------------------------------------------------------
--------
RecPro                     2.0772      0.180     11.571      0.000       1.725
2.429
ThemePro                  -2.5591      0.175    -14.616      0.000      -2.902
-2.216
LogLengthOfRecipient      -0.9592      0.087    -11.072      0.000      -1.129
-0.789
LogLengthOfTheme           0.6892      0.059     11.702      0.000       0.574
0.805
Dummy                      0.1922      0.159      1.211      0.226      -0.119
0.503
====================================================================================
========
```

[ ]:
```python
eval_model(model_4, test_set, predictors=["RecPro", "ThemePro",␣
 ↪"LogLengthOfRecipient", "LogLengthOfTheme", "Dummy"])
```

```
model accuracy on the test set: 0.861
model log-likelihood on the test set: -312.804
```

TODO: interpretation goes here.

It appears that the first model has a higher accuracy on the test set than the second one (0.864 compared to 0.861 respectively). However, the first has a lower log-likelihood on the test set than the second one (-315.059 compared to -312.804 respectively).

Thus, according to the first metric (accuracy), the first model is better. On the other hand, according to the second metric (log-likelihood) the second model is the better one.

Therefore, we will choose a model according to the metric which represents our objective. However, notice that both metrics are very similar for the two models. Thus, both models will give us similar results.

0.1.5 Part 5

For the theoretical discussion of the regression coefficients, we will interpret the coefficients of the first model (non-log model).

1. It appears that $\beta_{RecPro} = 2.455$. Hence we can conlcude the following:

- Since $\beta_{RecPro} > 0$, the probability of the dative alternation being DO is higher when the recipient (in the sentence) is a pronoun

  ($x_{RecPro} = 1$).

- The odds-ratio is $e^{2.455}$ times larger, when the recipient (in the sentence) is a pronoun ($x_{RecPro} = 1$).

2. It appears that $\beta_{ThemePro} = -2.766$. Hence we can conlcude the following:

- Since $\beta_{ThemePro} < 0$, the probability of the dative alternation being DO is lower when the theme (in the sentence) is a pronoun

($x_{ThemePro} = 1$).

- The odds-ratio is $e^{-2.766}$ times smaller, when the recipient (in the sentence) is a pronoun ($x_{ThemePro} = 1$).

3. It appears that $\beta_{RecLen} = -0.406$. Hence we can conculde the following:

- Since $\beta_{RecLen} < 0$, the probability of the dative alternation being DO decreases as the length of the recipient (in the sentence) increases.
- The odds-ratio is $e^{-0.406}$ times smaller, when the length of the recipient (in the sentence) increases by one.

4. It appears that $\beta_{ThemeLen} = 0.334$. Hence we can conculde the following:

- Since $\beta_{ThemeLen} > 0$, the probability of the dative alternation being DO increases as the length of the theme (in the sentence) increases.
- The odds-ratio is $e^{0.334}$ times larger, when the length of the theme (in the sentence) increases by one.

(*) As mentioned in the Pset PDF, we don't need to dicsuss the intercept (since its value dependes on the numeric cooding scheme used for the predictors).

Simplifying The Model Now, we will create a simplified model - which contains less explaining variables than the non-log full model.

In order to simplify the model and reduce the number of variables, we decided to combine the variables "LengthOfRecipient" and "LengthOfTheme" into a new variable "SqrtLengthSum", as follows:

$SqrtLengthSum = (LengthOfRecipient + LengthOfTheme)^{\frac{1}{2}}$

(Then, we removed the variables "LengthOfRecipient" and "LengthOfTheme").

Why we decided to create this variable:

Even though we lose information on the individual lengths of the recipient and the theme, the sum still holds some information regarding their lengths. In addition, we decided to take the square root of the sum, in order to scale it down.

[ ]:
```
train_set["SqrtLengthSum"] = np.sqrt(train_set["LengthOfRecipient"] +␣
 ↪train_set["LengthOfTheme"])
test_set["SqrtLengthSum"] = np.sqrt(test_set["LengthOfRecipient"] +␣
 ↪test_set["LengthOfTheme"])

X = train_set[["RecPro", "ThemePro", "SqrtLengthSum", "Dummy"]]
y = train_set[["Response"]]

model_5 = sm.GLM(y, X, family=sm.families.Binomial())
model_5 = model_5.fit()
print(model_5.summary())
```

```
                 Generalized Linear Model Regression Results
==============================================================================
```

```
Dep. Variable:               Response   No. Observations:                 2610
Model:                            GLM   Df Residuals:                     2606
Model Family:                Binomial   Df Model:                            3
Link Function:                  Logit   Scale:                          1.0000
Method:                          IRLS   Log-Likelihood:                -1030.7
Date:                Sat, 13 May 2023   Deviance:                       2061.5
Time:                        19:21:49   Pearson chi2:                 2.58e+03
No. Iterations:                     6   Pseudo R-squ. (CS):             0.2937
Covariance Type:            nonrobust
================================================================================
=
                   coef    std err          z      P>|z|      [0.025
0.975]
--------------------------------------------------------------------------------
-
RecPro           3.1146      0.150     20.814      0.000       2.821
3.408
ThemePro        -2.8693      0.170    -16.881      0.000      -3.202
-2.536
SqrtLengthSum    0.3912      0.072      5.465      0.000       0.251
0.532
Dummy           -0.8537      0.208     -4.112      0.000      -1.261
-0.447
================================================================================
=
```

[ ]:

```
eval_model(model_5, test_set, predictors=["RecPro", "ThemePro",␣
 ↪"SqrtLengthSum", "Dummy"])
```

```
model accuracy on the test set: 0.795
model log-likelihood on the test set: -374.076
```

As we can see, the new model's accuracy is 0.795, compared to 0.864 by the full non-log model. Thus, the predictive accuracy sacrificed is $0.069 < 0.1$. Therefore, the accuracy lost is considered "not much" (according to the forums), as required.

Improving Our Solution We will now try another model in order to improve the accuracy of the previous suggested model.

Here, we decided to combine the variables "RecPro" and "ThemePro" into a new variable "SubPro", as follows:

$SubPro = RecPro - ThemePro$

(Then, we removed the variables "RecPro" and "ThemePro").

Why we decided to create this variable:

Notice that the values of the new variable 'SubPro' are: (-1, 0, 1).

Moreover, the following holds:

- $SubPro = 1$ IFF $RecPro = 1$ and $ThemePro = 0$.
- $SubPro = -1$ IFF $RecPro = 0$ and $ThemePro = 1$.
- $SubPro = 0$ IFF ($RecPro = ThemePro = 0$ or $RecPro = ThemePro = 1$).

We can see that information loss happens only when $SubPro = 0$ (two different cases are represented by a single value).

However, In the full non-log model, it appears that $\beta_{RecPro} = 2.455$ and $\beta_{ThemePro} = -2.766$.

Thus, we can see that $\beta_{RecPro} \approx -\beta_{ThemePro}$.

As a result, the probability of a sentence with both recipient and theme as pronouns being classified as DO, is approximately the same as the probability of a sentence with none as pronouns being classified as DO. In other words, the case where $RecPro = ThemePro = 0$ is roughly the same as the case where $RecPro = ThemePro = 1$ (according to the model).

Therefore, the new variable 'SubPro' successfully encodes both variables 'RecPro' and 'ThemePro' without losing a significant amount of information.

[ ]:
```
train_set["SubPro"] = train_set["RecPro"] - train_set["ThemePro"]
test_set["SubPro"] = test_set["RecPro"] - test_set["ThemePro"]

X = train_set[["SubPro", "LengthOfRecipient", "LengthOfTheme", "Dummy"]]
y = train_set[["Response"]]

model_5_improved = sm.GLM(y, X, family=sm.families.Binomial())
model_5_improved = model_5_improved.fit()
print(model_5_improved.summary())
```

```
                 Generalized Linear Model Regression Results
==============================================================================
Dep. Variable:               Response   No. Observations:                 2610
Model:                            GLM   Df Residuals:                     2606
Model Family:                Binomial   Df Model:                            3
Link Function:                  Logit   Scale:                          1.0000
Method:                          IRLS   Log-Likelihood:                -897.91
Date:                Sat, 13 May 2023   Deviance:                       1795.8
Time:                        19:21:49   Pearson chi2:                 3.51e+03
No. Iterations:                     6   Pseudo R-squ. (CS):             0.3620
Covariance Type:            nonrobust
================================================================================
=====
                        coef    std err          z      P>|z|      [0.025
0.975]
--------------------------------------------------------------------------------
-----
SubPro                2.5897      0.142     18.201      0.000       2.311
2.869
LengthOfRecipient    -0.3800      0.039     -9.823      0.000      -0.456
-0.304
```

```
LengthOfTheme          0.2416      0.022     10.764      0.000       0.198
0.286
Dummy                  0.1593      0.117      1.356      0.175      -0.071
0.389
================================================================================
=====
```

[ ]:
```
eval_model(model_5_improved, test_set, predictors=["SubPro",␣
↪"LengthOfRecipient", "LengthOfTheme", "Dummy"])
```

```
model accuracy on the test set: 0.85
model log-likelihood on the test set: -314.611
```

As we can see, the new model's accuracy is 0.85, compared to 0.864 by the full non-log model.

Thus, the predictive accuracy sacrificed is 0.014 (roughly 5 times less than the previous suggested model).

Therefore, we succeeded in our task to improve our simplifed model.