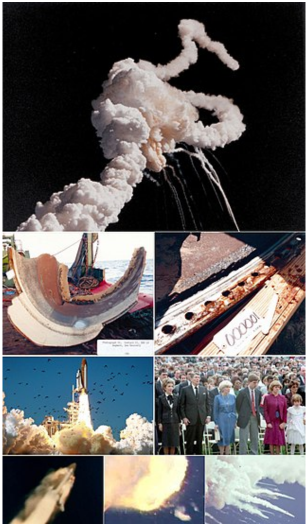
Machine Learning and Applications: Logistic Regression

Alessio Benavoli

CSIS
University of Limerick

The Challenger Disaster



On January 28, 1986, the NASA Space Shuttle Challenger broke apart 73 seconds into its flight

The Challenger Disaster

NASA appointed members of the Rogers Commission to investigate the cause of the disaster.

The commission determined that the disaster began with the failure of an O-ring seal in the solid rocket motor due to the cold temperature (-0.6 Celsius degrees) during the launch.

The problem with O-rings was something known: the night before the launch, there was a three-hour teleconference between NASA engineers, discussing the effect of low temperature forecasted for the launch on the O-ring performance.

Whole dataset

Temp	Damage_Incident
66	0.0
70	1.0
69	0.0
68	0.0
67	0.0
72	0.0
73	0.0
70	0.0
57	1.0
63	1.0
70	1.0
78	0.0
67	0.0
53	1.0
67	0.0
75	0.0
70	0.0
81	0.0
76	0.0
79	0.0
75	1.0
76	0.0
58	1.0

The Challenger Disaster

The conclusion of NASA engineers based on the data below was

Temperature data is not conclusive on predicting primary O-ring failure

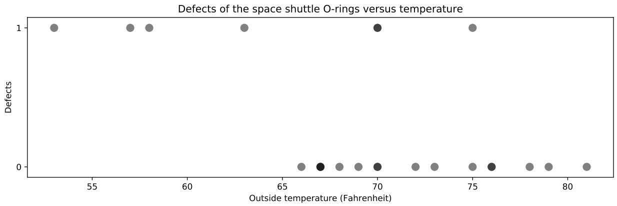
In fact the failures seem to appear at any temperature.

Temp	Damage_Incident
70	1.0
57	1.0
63	1.0
70	1.0
53	1.0
75	1.0
58	1.0

Rogers Commission

The Rogers Commission observed a major flaw in this conclusion:

> *the flights with zero incidents were excluded from the analysis because it was felt that these flights did not contribute any information about the temperature effect. However, the whole dataset clearly shows the correlation of O-ring damage in low temperature.*



Our goal

Our goal is to address the following questions:

1. Is the temperature associated with O-ring incidents?
2. How was the temperature affecting the probability of O-ring incidents?
3. What was the predicted probability of an incident in an O-ring for the temperature of the launch day?

The input variable in this case is

- temperature (continuous)

The output variable is

- failure (categorical/binary)

Therefore, this is a classification problem. Given a value of the temperature we want to classify O-rings according to two classes:

- O-ring will fail
- O-ring won't fail

More specifically, we want to determine the probability of the two classes: p(O-ring will fail), p(O-ring won't fail)=1-p(O-ring will fail)

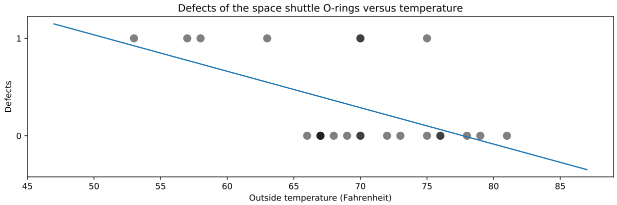
Can we use linear regression?

We could use linear regression and model:

$$failure = \alpha + \beta t$$

where failure can be either 0 and 1.

This is the resulting linear regression line:



It is not a good model, because we are not considering the fact that failure is a categorical variable.

Model

We need a “regression model”, that is a function of temperature but that is bounded between 0 and 1 and varies between 0 and 1 as the temperature changes.

We can squeeze our linear regression line through a non-linear function that bounds the output between 0 and 1:

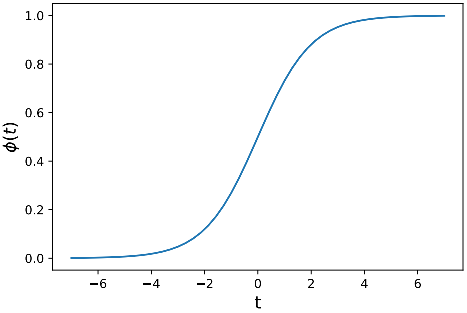
$$\phi(\alpha + \beta t) = \frac{1}{1 + e^{-\alpha - \beta t}}$$

Logistic function

For instance for $\alpha = 0, \beta = 1$, the shape of the logistic function

$$\phi(t) = \frac{1}{1 + e^{-t}}$$

is



We could then interpret:

- $\phi(\alpha + \beta t)$ is the probability of failure

For instance from the plot, the probability of dailure at temperature 0 is 0.5, at temperature 6 is 1 etc..

Logistic regression likelihood

- $\phi(\alpha + \beta t)$ is the probability of failure then
- $1 - \phi(\alpha + \beta t)$ is the probability of no-failure

We can rewrite the above if as

$$p(y|t, \alpha, \beta) = \phi(\alpha + \beta t)^y (1 - \phi(\alpha + \beta t))^{1-y} \text{ for } y = 0, 1$$

This is the likelihood that an observation y is either equal to 0 (no-failure) or 1 (failure) when the probability of failure is $\phi(\alpha + \beta t)$.

It is a **Bernoulli distribution**, same distribution we used for MultinomialNB for a binary class.

We have N=23 observations in our dataset. If we assume the observations are conditionally independent given the parameters α, β then

$$\begin{aligned} p(\mathcal{D}|\alpha, \beta) &:= p(y_1, \ldots, y_N | t_1, \ldots, t_N, \alpha, \beta) \\ &= \prod_{(t_i, y_i) \in \mathcal{D}} \phi(\alpha + \beta t_i)^{y_i} (1 - \phi(\alpha + \beta t_i))^{1-y_i} \end{aligned}$$

Fitting: Maximum Likelihood Estimation

We aim to find the best parameters α, β that fit the data by maximising the likelihood function

$$\arg\max_{\alpha,\beta} \prod_{(t_i,y_i)\in\mathcal{D}} \phi(\alpha+\beta t_i)^{y_i}(1-\phi(\alpha+\beta t_i))^{1-y_i}$$

Contrarily to MultinomialNB, there is no analytical solution, we need to solve that optimisation numerically:

```
t = challenger_data.iloc[:,0].values #temperatures
y = challenger_data.iloc[:,1].values #failures
X = t.reshape(-1,1)
from sklearn.linear_model import LogisticRegression
#fit the model (learning)
clf = LogisticRegression(solver='lbfgs')
clf.fit(X, y)
```

“lbfgs” is the name of the numerical optimisation algorithm that is used to find MLE.

Predictions

Optimal parameters

```
print(clf.intercept_,clf.coef_)#alpha, beta
>>14.862, -0.23
```

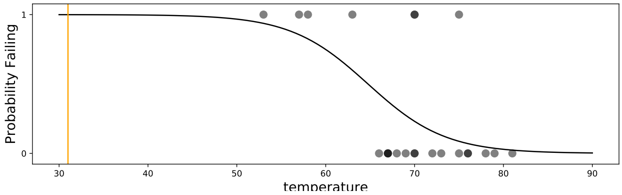
We can now plot the fitting line

```
tpred= np.linspace(31,90,100)
Xpred = tpred.reshape(-1,1)
plt.figure(figsize=(12.5, 3.5))

plt.plot(Xpred,1/(1+np.exp(-clf.intercept_-clf.coef_*Xpred)), color=
                                  "k")

plt.xlabel("temperature",fontsize=16);
plt.ylabel("Probability Failing",fontsize=16,rotation=90)
plt.scatter(challenger_data.iloc[:, 0], challenger_data.iloc[:, 1],
                                  s=75,
 color="k", alpha=0.5)
plt.yticks([0, 1]);
```

Logistic line



The line gives the probability of failure as a function of the temperature.

For instance, at the temperature of the launch 31F, the probability of failure is

$$\phi(14.83 - 0.23 \cdot 31) = \frac{1}{1 + e^{-14.83 + 0.23 \cdot 31}} \approx 1$$

We can conclude that the disaster could be avoided!

```
 print("Predicted prob of [no_fail, fail]=",clf.predict_proba(np.
                                    array([[31]])))
print("Predicted class=",clf.predict(np.array([[31]])))
>>[0,1]
>>[1]
```

Regularisation (smoothing)

In MultinomialNB, we used “Laplace smoothing” (α) to introduce some regularisation.

We can do the same in Logistic regression. In sklearn, the regularisation parameter for Logistic regression is called C.

Its default value is 1: BE AWARE that smaller values of C means stronger regularisation, so you think of $C = 1/\alpha$.

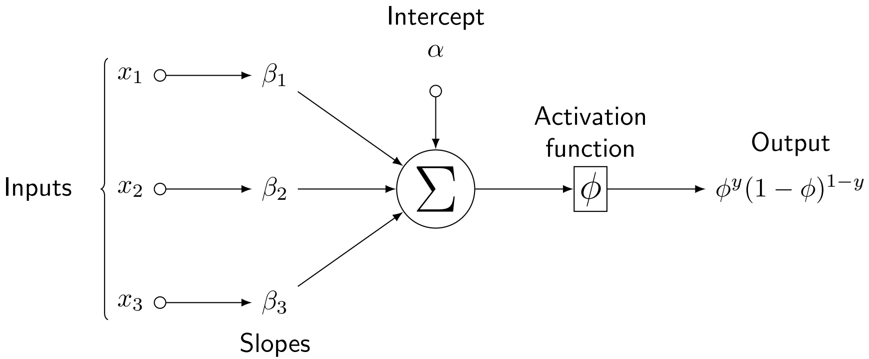
```
clf = LogisticRegression(solver='lbfgs', C=1)
```

Rule-of-thumb: use $C = 1$ or select the optimal-value of C via 10-fold cross-validation.

Logistic Regression with multiple inputs

In case, we have three (or more) input variables we can generalize logistic regression as

$$\phi(\alpha + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3) = \frac{1}{1 + e^{-\alpha - \beta_1 x_1 - \beta_2 x_2 - \beta_3 x_3}}$$



The output $\phi^y(1-\phi)^{1-y}$ means we observe $y = 1$ with probability ϕ and $y = 0$ with probability $1 - \phi$

Logistic Regression

We can also use logistic regression when we have more than two classes (e.g., 0 no-fail, 1 fail, 2 red-light).

We use the option multi_class='multinomial'

```
clf = LogisticRegression(solver='lbfgs', multi_class='multinomial')
```

Credit Card competition

Business Analytics: Banks receives so many applications for credit card (CC) request.

Going through each request manually can be very time consuming, also prone to human errors.

However, if we can use the historical data to build a model which can shortlist the candidates for approval that save time.

The provided training dataset contains past applications for CC. Each row in the dataset includes information on the applicant (sex,age,Debt, Married etc.) and the output is the decision (1: approved, 0: denied).

Goal: by using any ML classifier, you must build a predictor model that predicts if a CC should be approved for a person that has certain characteristics (sex,age,Debt, Married etc.)

Credit Card competition

See accompanying notebook