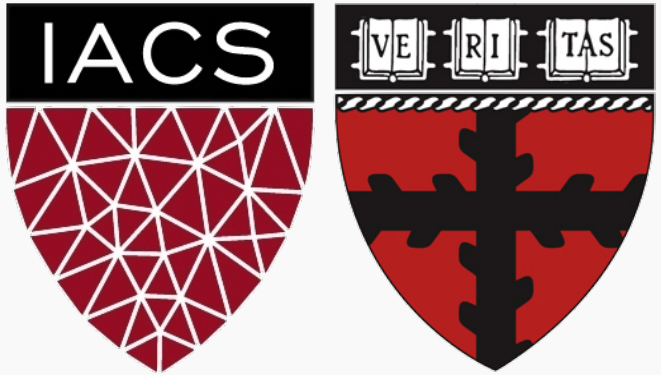
Lecture 15: Logistic Regression II

CS109A Introduction to Data Science

Pavlos Protopapas, Natesh Pillai



Lecture Outline

- Interpreting interactions in logistic regression
- Classification Boundaries
- Regularization in Logistic Regression
- Bayes Theorem and Misclassification Rates
- ROC Curves

Interactions in Multiple Logistic Regression

Just like in linear regression, interaction terms can be considered in logistic regression. An **interaction terms** is incorporated into the model the same way, and the interpretation is very similar (on the log-odds scale of the response of course).

Write down the model for the Heart data for the 2 predictors plus the interactions term based on the output on the next slide.

Interpreting Multiple Logistic Regression: an Example

The results for the multiple logistic regression model are:

```
df_heart['Age_Sex'] = df_heart.Age*df_heart.Sex

data_x = df_heart[['Age','Sex','Age_Sex']]

logit3 = LogisticRegression(penalty='none', max_iter = 1000)
logit3.fit(data_x, df_heart['AHD'])

print("Logistic Regression Estimated Betas (B0,B1,B2,B3):",
      np.round(logit3.intercept_,4),np.round(logit3.coef_,5))
```

```
Logistic Regression Estimated Betas (B0,B1,B2,B3): [-4.2742] [[0.05654 0.77549 0.01273]]
```

Some questions

```
Logistic Regression Estimated Betas (B0,B1,B2,B3): [-4.2742] [[0.05654 0.77549 0.01273]]
```

1. Write down the complete model. Break this down into the model to predict log-odds of heart disease (HD) based on Age for women and the same model for men.

2. Interpret the results of this model. What does the coefficient for the interaction term represent?

Classification Boundaries

Classification boundaries

Recall that we could attempt to purely classify each observation based on whether the estimated $P(Y = 1)$ from the model was greater than 0.5.

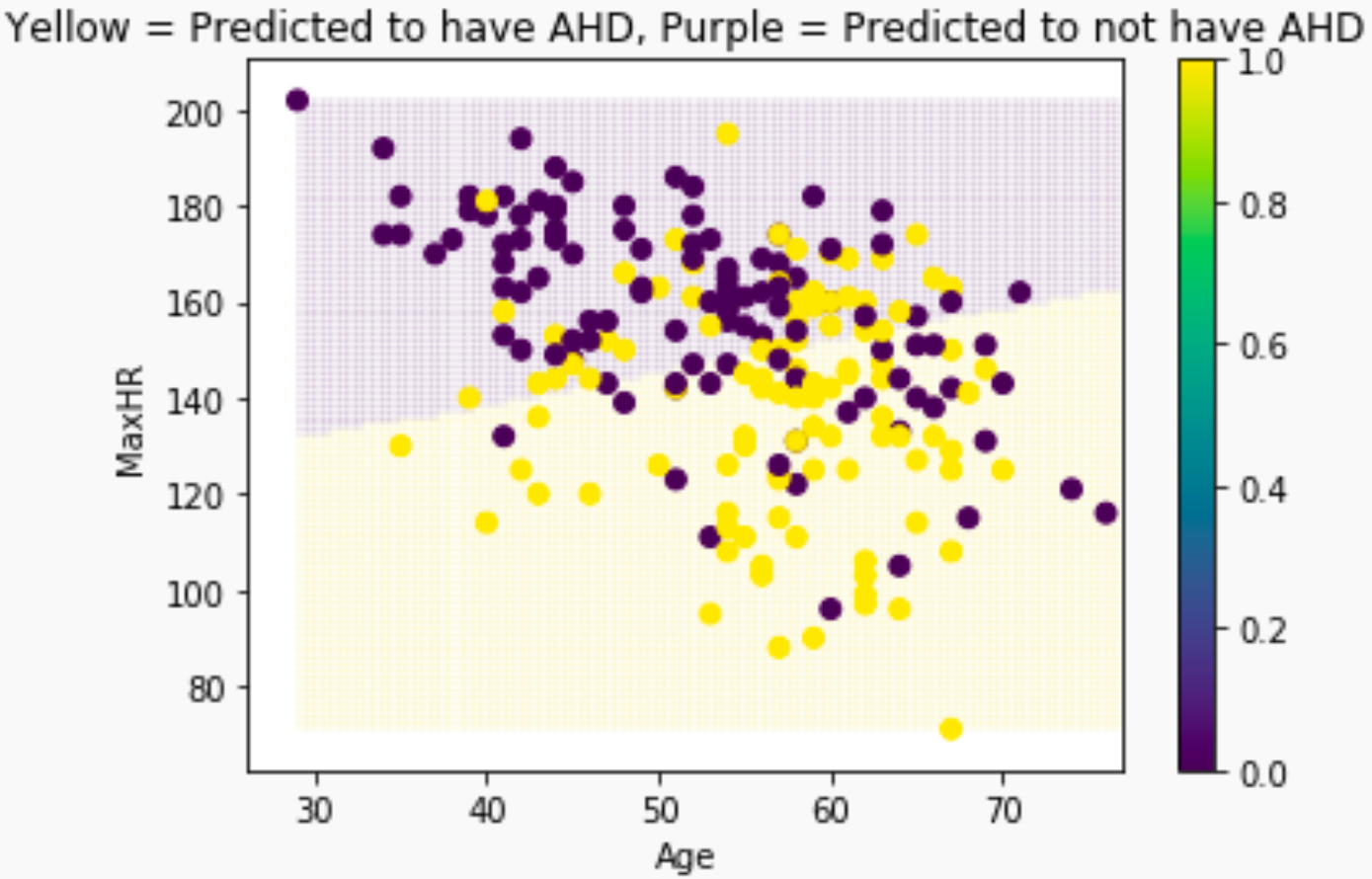
Recall, the logistic regression model statement from last class:

$$\ln\left(\frac{P(Y=1)}{P(Y=0)}\right) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$$

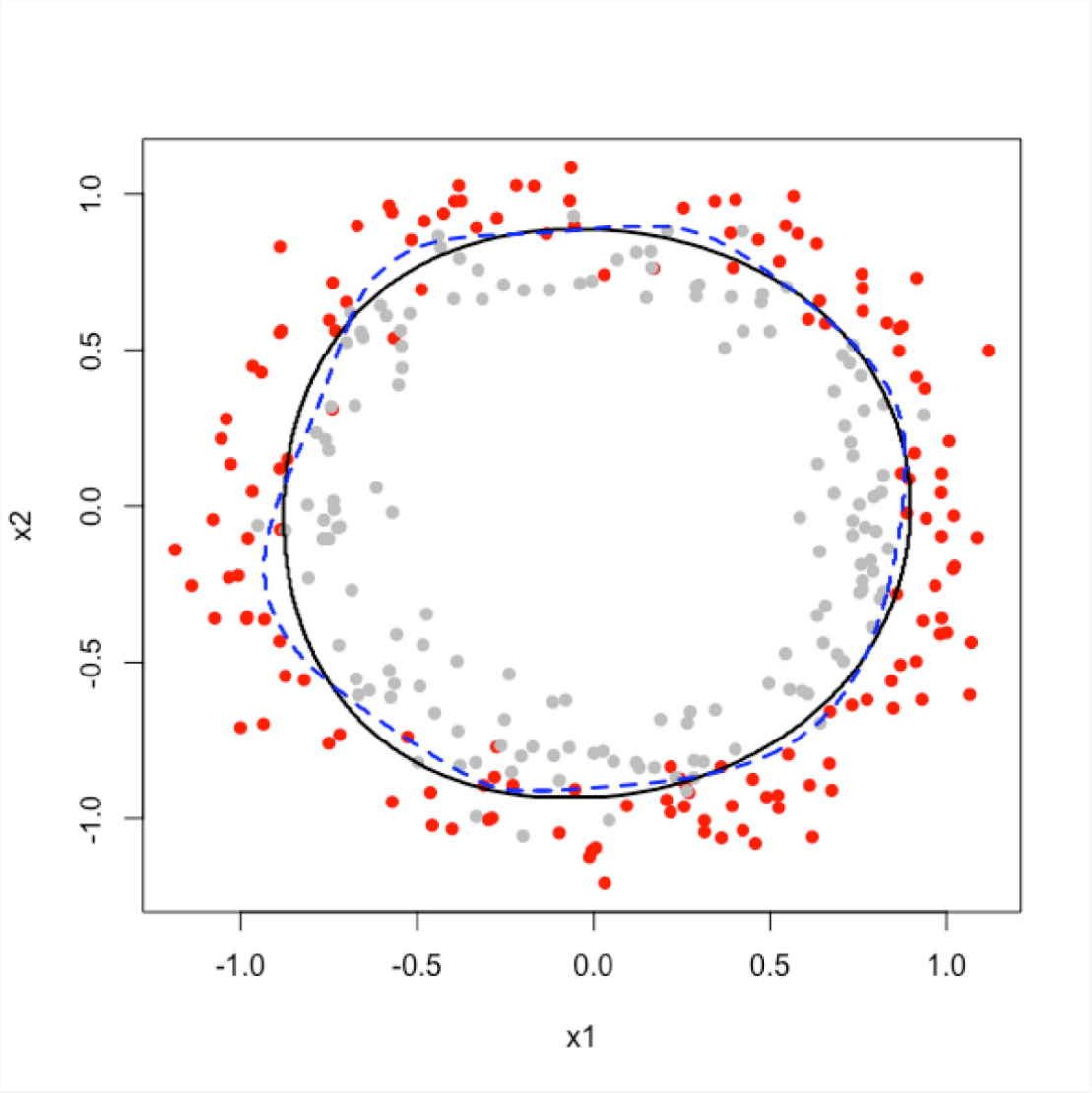
If $P(Y = 1) = 0.5$, then the odds is 1, and the log-odds is 0. Thus the classification boundary that separates the estimated probabilities above 0.5 from below 0.5 is defined when we set: $\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p = 0$.

Thus logistic regression is said to lead to a **linear classification boundary.** There is actually a lot more freedom in logistic regression's classification boundary (the geometry is defined by the parameterization of your predictors!)

Linear Classification Boundary Example



Geometry of 2D Classification in Logistic Regression: an Example



2D Classification in Logistic Regression: an Example

Would a logistic regression model perform well in classifying the observations in this example?

What would be a good logistic regression model to classify these points?

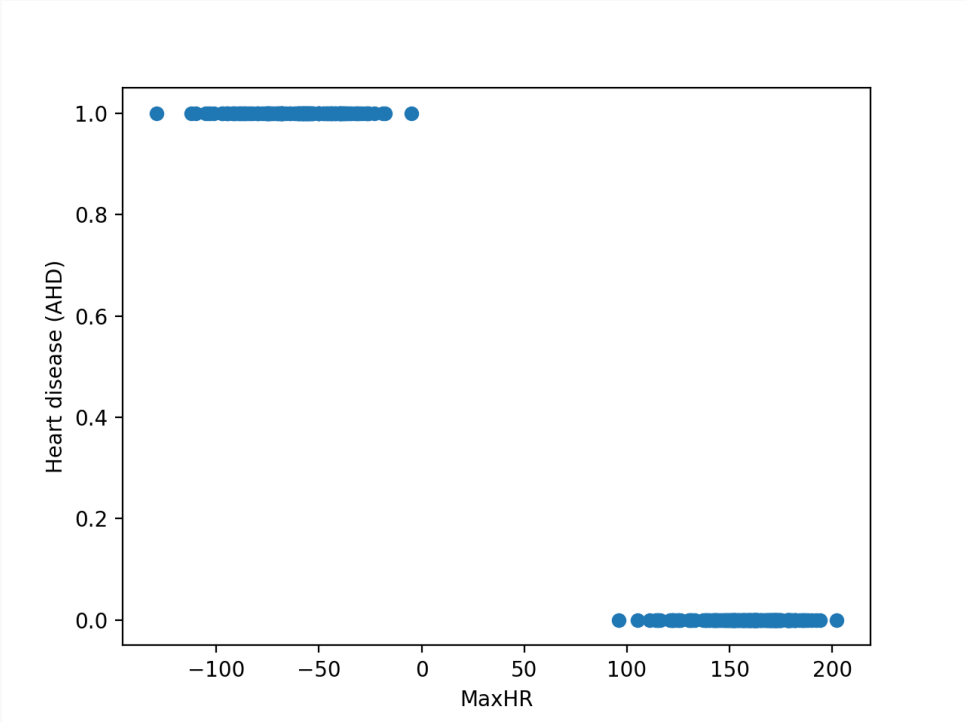
Based on these predictors, two separate logistic regression model were considered that were based on different ordered polynomials of X_1, X_2 and their interactions. The 'circles' represent the boundary for classification.

How can the classification boundary be calculated for a logistic regression?

Regularization in Logistic Regression

Motivation: MLE may not Exist

The problem of separation arises when the binary responses can be separated into all 0s and all 1s as a linear function of the predictors.



Motivation: MLE may not Exist

MLE does not exist when the data is perfectly separable.

```
logreg = sm.Logit(data_y, data_x).fit()
print(logreg.summary())
```

```
---------------------------------------------------------------------------
PerfectSeparationError                    Traceback (most recent call last)
<ipython-input-7-f13c7b317e38> in <module>
----> 1 logreg = sm.Logit(data_y, data_x).fit()
      2 print(logreg.summary())
```

The likelihood always increases with the magnitude of the estimated coefficients.

Need regularization!

Review: Regularization in Linear Regression

Based on the Likelihood framework, a loss function can be determined based on the log-likelihood function.

We saw in linear regression that maximizing the log-likelihood is equivalent to minimizing the sum of squares error

$$\arg\min \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \arg\min \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 x_{1i} + ... + \beta_p x_{pi}))^2$$

Review: Regularization in Linear Regression

And a regularization approach was to add a penalty factor to this equation. Which for Ridge Regression becomes:

$$\arg\min \left[\sum_{i=1}^{n} \left(y_i - \left(\beta_0 + \sum_{j=1}^{n} \beta_j x_{ji} \right) \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right]$$

Note: this penalty *shrinks* the estimates towards zero, and had the analogue of using a Normal prior centered at zero in the Bayesian paradigm.

Recall: Loss function in Logistic Regression

A similar approach can be used in logistic regression. Here, maximizing the log-likelihood is equivalent to minimizing the following loss function:

$$\underset{\beta_0,\beta_1,\ldots,\beta_p}{\operatorname{argmin}} \left[-\sum_{i=1}^{n} (y_i \ln(p_i) + (1-y_i)\ln(1-p_i)) \right]$$

where $p_i = \frac{1}{1-e^{-(\beta_0+\beta_1 X_{1,i}+\cdots+\beta_p x_{p,i})}}$

Why is this a good loss function to minimize? Where does this come from?

The log-likelihood for independent $Y_i \sim \text{Bern}(p_i)$.

Regularization in Logistic Regression

A penalty factor can then be added to this loss function and results in a new loss function that penalizes large values of the parameters:

$$\underset{\beta_0,\beta_1,\ldots,\beta_p}{\operatorname{argmin}} \left[-\sum_{i=1}^{n} (y_i \ln(p_i) + (1-y_i)\ln(1-p_i)) + \lambda \sum_{j=1}^{p} \beta_j^2 \right]$$

The result is just like in linear regression: shrink the parameter estimates towards zero.

In practice, the intercept is usually not part of the penalty factor.

Note: the sklearn package uses a different tuning parameter: instead of λ they use a constant that is essentially $C = \frac{1}{\lambda}$.

Regularization in Logistic Regression: an Example

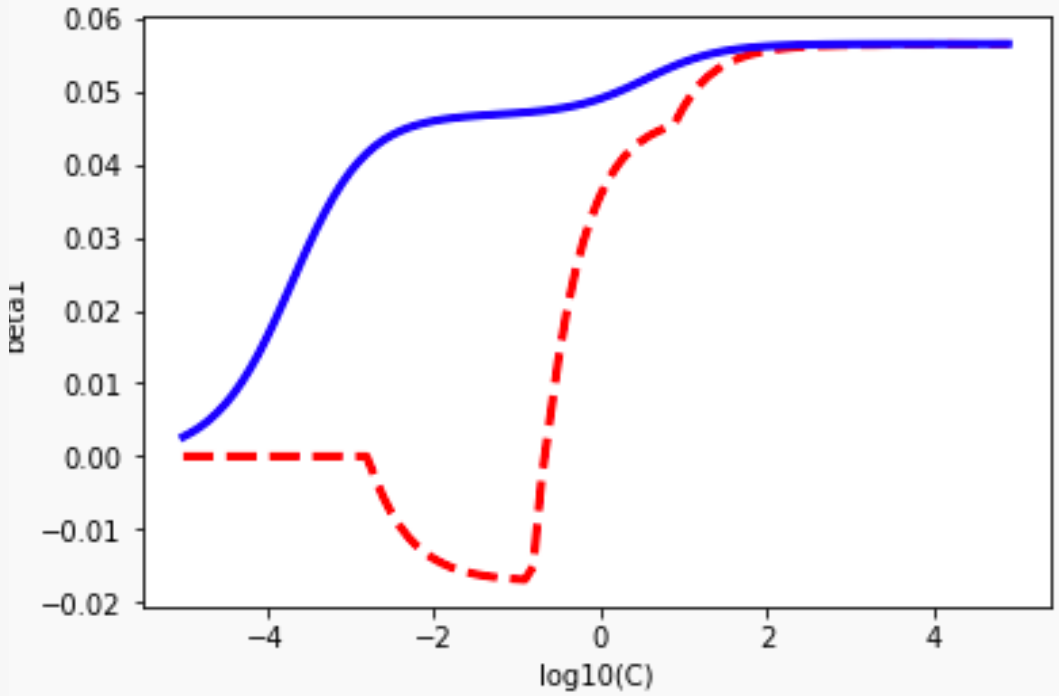
Let's see how this plays out in an example in logistic

```
beta1_l1 = []
beta1_l2 = []
Cs = np.power(10,np.arange(0,10,0.1))/np.power(10,5)

data_x = df_heart[['Age','Sex','Age_Sex']]
data_y = df_heart['AHD']

for C in Cs:
    logitm_l1 = LogisticRegression(C = C, penalty = "l1",
                        solver='liblinear',max_iter=200)
    logitm_l1.fit (data_x, data_y)
    logitm_l2 = LogisticRegression(C = C, penalty = "l2",
                        solver='lbfgs', max_iter=200)
    logitm_l2.fit (data_x, data_y)
    beta1_l1.append(logitm_l1.coef_[0][0])
    beta1_l2.append(logitm_l2.coef_[0][0])

plt.plot(np.log10(Cs), beta1_l1, "--", color='red', lw=3)
plt.plot(np.log10(Cs), beta1_l2, color='blue', lw=3)
plt.xlabel ("log10(C)")
plt.ylabel("beta1")
plt.show()
```

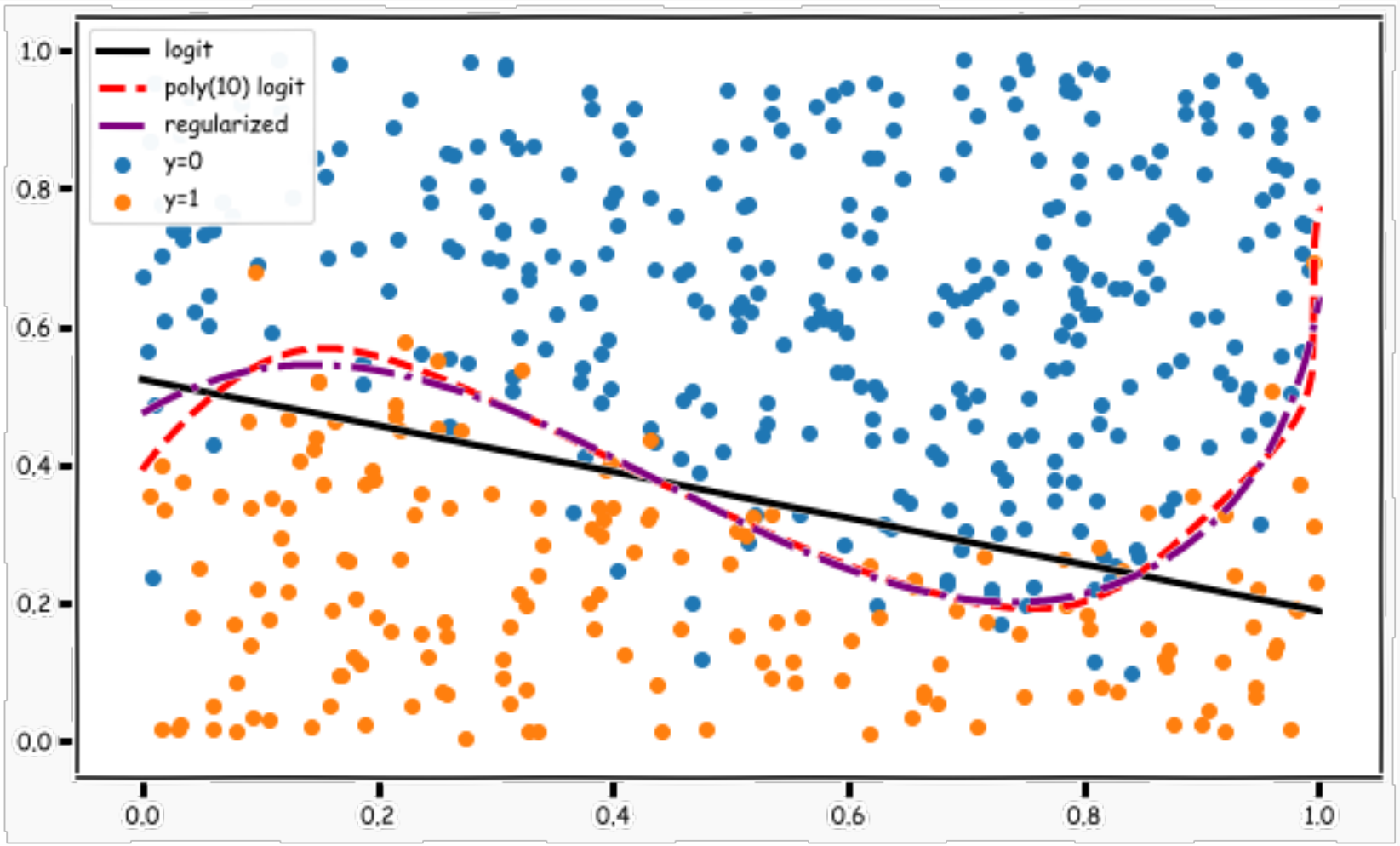


Regularization in Logistic Regression: tuning λ

Just like in linear regression, the shrinkage factor must be chosen. How should we go about doing this?

Through building multiple training and test sets (hooray, cross-validation!), we can select the best shrinkage factor to mimic out-of-sample prediction.

Regularized Decision Boundaries



Bayes Theorem, Misclassification Rates, False Positives and Negatives

Bayes' Theorem

We defined conditional probability as:

$$P(B|A) = P(B \text{ and } A)/P(A)$$

And using the fact that $P(B \text{ and } A) = P(A|B)P(B)$ we get the simplest form of Bayes' Theorem:

$$P(B|A) = \frac{P(A|B)P(B)}{P(A)}$$

Another version of Bayes' Theorem is found by substituting in the Law of Total Probability (LOTP) into the denominator:

$$P(B|A) = \frac{P(A|B)P(B)}{P(A|B)P(B) + P(A|B^C)P(B^C)}$$

Where have we seen Bayes' Theorem before? Why do we care?

Diagnostic Testing

In the diagnostic testing paradigm, one cares about whether the results of a test (like a classification test) matches truth (the true class that observation belongs to). The simplest version of this is trying to detect disease ($D+$ vs. $D-$) based on a diagnostic test ($T+$ vs. $T-$).

Diagnostic Testing

Medical examples include various screening tests: breast cancer screening through (i) self-examination and (ii) mammography, prostate cancer screening through (iii) PSA tests, and Colo-rectal cancer through (iv) colonoscopies.

These tests are a little controversial because of poor predictive probability of the tests.

Diagnostic Testing (cont.)

Bayes' theorem can be rewritten for diagnostic tests:

$$P(D+|T+) = \frac{P(T+|D+)P(D+)}{P(T+|D+)P(D+) + P(T+|D-)P(D-)}$$

These probability quantities can then be defined as:

- *Sensitivity*: $P(T+|D+)$
- *Specificity*: $P(T-|D-)$
- *Prevalence*: $P(D+)$
- *Positive Predictive Value*: $P(D+|T+)$
- *Negative Predictive Value*: $P(D-|T-)$

How do positive and negative predictive values relate? Be careful…

Diagnostic Testing

We mentioned that these tests are a little controversial because of their poor predictive probability. When will these tests have poor positive predictive probability?

When the disease is not very prevalent, then the number of 'false positives' will overwhelm the number of true positive. For example, PSA screening for prostate cancer has sensitivity of about 90% and specificity of about 97% for some age groups (men in their fifties), but prevalence is about 0.1%.

What is positive predictive probability for this diagnostic test?

Why do we care?

As data scientists, why do we care about diagnostic testing from the medical world?

Because classification can be thought of as a diagnostic test. Let $Y_i = k$ be the event that observation i truly belongs to category k, and let $\hat{Y}_i = k$ the event that we correctly predict it to be in class k. Then Bayes' rule states that our *Positive Predictive Value* for classification is:

$$P(Y_i = k|\hat{Y}_i = k) = \frac{P(\hat{Y}_i = k|Y_i = k)P(Y_i = k)}{P(\hat{Y}_i = k|Y_i = k)P(Y_i = k) + P(\hat{Y}_i = k|Y_i \neq k)P(Y_i \neq k)}$$

Thus the probability of a predicted outcome truly being in a specific group depends on what? The proportion of observations in that class!

Error in Classification

There are 2 major types of error in classification problems based on a binary outcome. They are:

False positives: incorrectly predicting $\hat{Y} = 1$ when it truly is in $Y = 0$.

False negative: incorrectly predicting $\hat{Y} = 0$ when it truly is in $Y = 1$.

The results of a classification algorithm are often summarized in two ways: (1) a **confusion matrix**, sometimes called a **contingency table**, or a 2x2 table (more generally (k x k) table) and (2) a receiver operating characteristics (ROC) curve.

Confusion matrix

When a classification algorithm (like logistic regression) is used, the results can be summarize in a (k x k) table as such:

	Predicted no AHD ($\hat{Y} = 0$)	Predicted AHD ($\hat{Y} = 1$)
Truly no AHD ($Y = 0$)	110	54
Truly AHD ($Y = 1$)	53	86

The table above was a classification based on a logistic regression model to predict AHD based on "3" predictors: X_1 = Age, X_2 = Sex, and X_3 = interaction between Age and Sex.

What are the false positive and false negative rates for this classifier?

Bayes' Classifier Choice

A classifier's error rates can be tuned to modify this table. How?

The choice of the Bayes' classifier level will modify the characteristics of this table.

If we thought is was more important to predict AHD patients correctly (fewer false negatives), what could we do for our Bayes' classifier level?

We could classify instead based on:

$$\hat{P}(Y = 1) > \pi$$

and we could choose π to be some level other than 0.50.

Let's see what the table looks like if π were 0.40 or 0.60 instead. What should happen to the False Positive and False Negative frequencies?

Other Confusion tables

Based on π = 0.4:

	Predicted no AHD ($\hat{Y} = 0$)	Predicted AHD ($\hat{Y} = 1$)
Truly no AHD ($Y = 0$)	93	71
Truly AHD ($Y = 1$)	38	101

What has improved? What has worsened?

Based on π = 0.6:

	Predicted no AHD ($\hat{Y} = 0$)	Predicted AHD ($\hat{Y} = 1$)
Truly no AHD ($Y = 0$)	138	26
Truly AHD ($Y = 1$)	74	65

Which should we choose? Why?

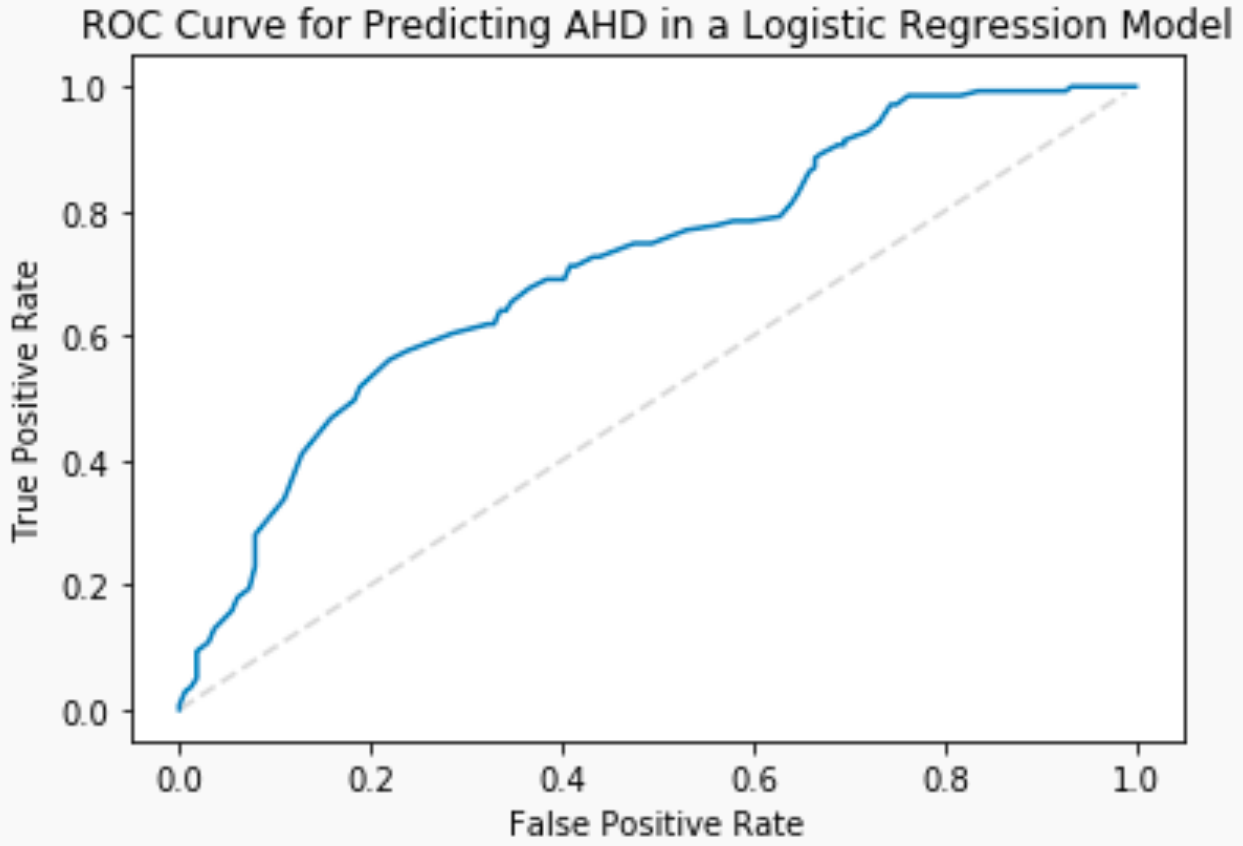
ROC Curves and Area-Under-the-Curve

ROC Curves

The Radio Operator Characteristics (ROC) curve illustrates the trade-off for all possible thresholds chosen for the two types of error (or correct classification).

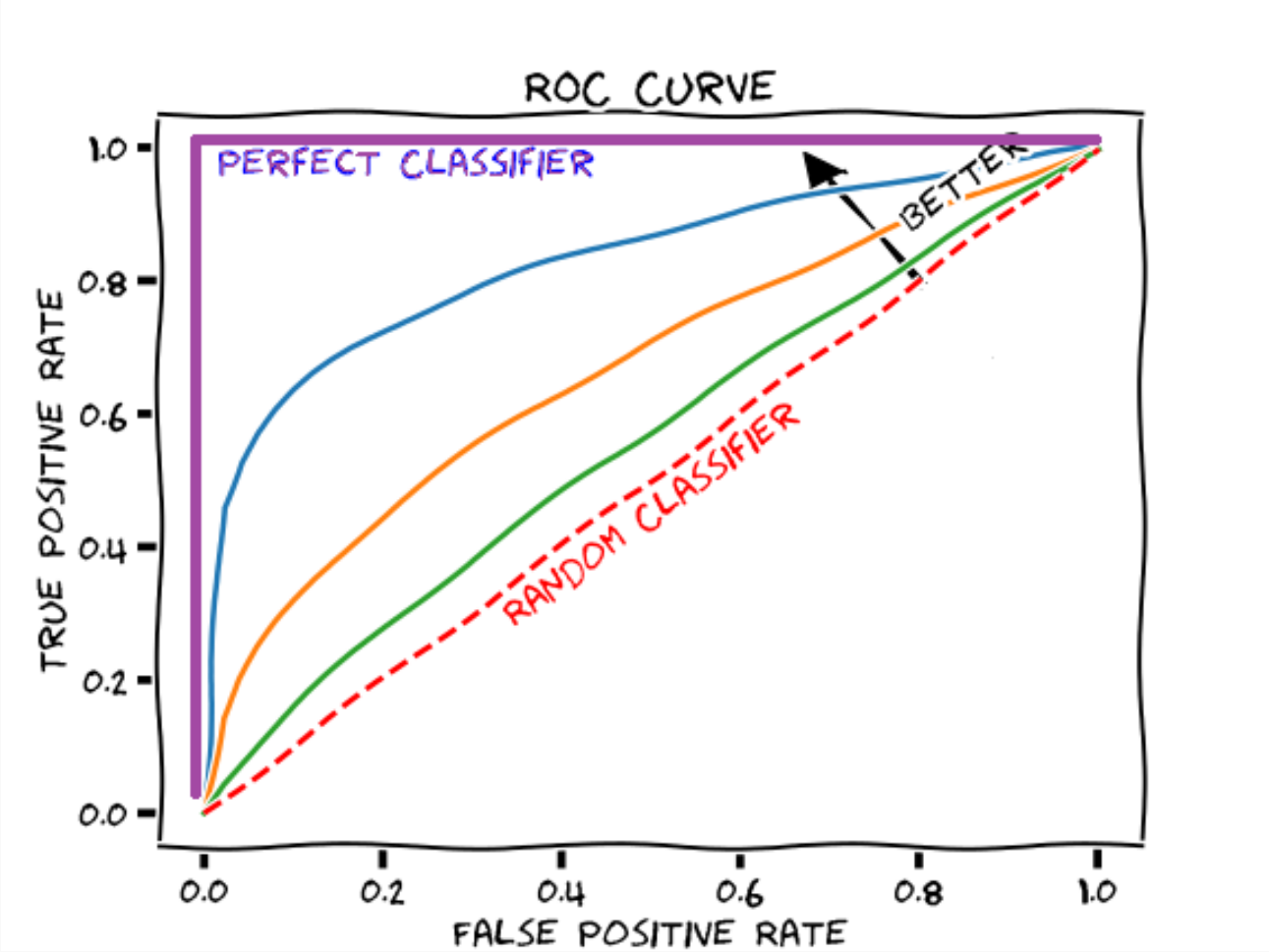
The vertical axis displays the true positive predictive value and the horizontal axis depicts the true negative predictive value.

ROC Curve Example



What is the shape of an ideal ROC curve?

ROC Curve Example



AUC for measuring classifier performance

The overall performance of a classifier, calculated over all possible thresholds, is given by the **area under the ROC curve** (AUC).

An ideal ROC curve will hug the top left corner, so the larger the AUC the better the classifier.

This AUC then can be use to compare various approaches to classification: Logistic regression, *k*-NN, Decision Trees (to come), etc.