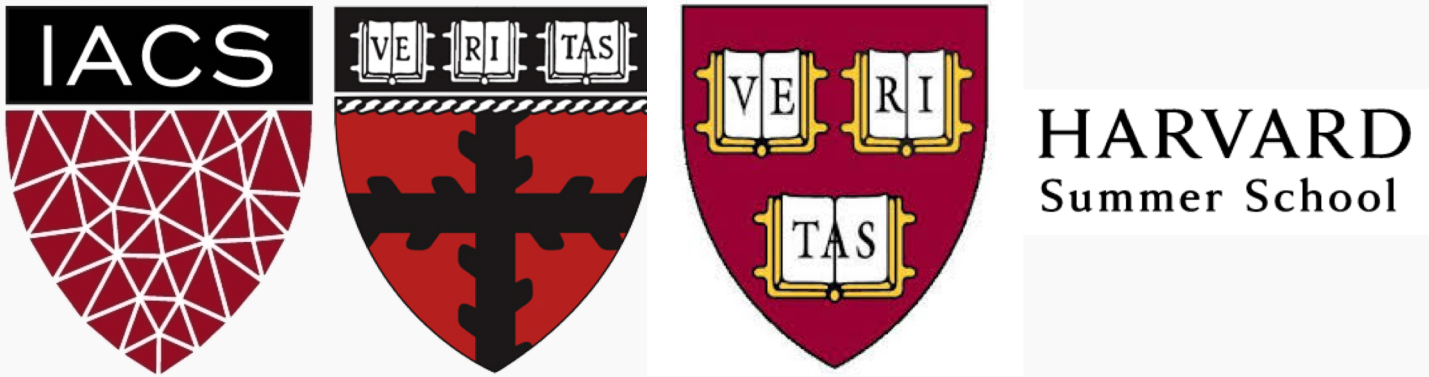
Lecture #6: Logistic Regression

CS-S109A: Introduction to Data Science

Kevin Rader



Lecture Outline

- Classification: Why not Linear Regression?
- Binary Response & Logistic Regression
  - Estimating the Simple Logistic Model
  - Classification using the Logistic Model
  - Multiple Logistic Regression
- Classification Boundaries
- Regularization in Logistic Regression
- Bayes Theorem and Misclassification Rates
- ROC Curves
- Multiclass Classification
  - Multinomial and OvR Logistic Regression
  - Multiclass *k*-NN

Advertising Data (from earlier lectures)

X
predictors
features
covariates

Y
outcome
response variable
dependent variable

TV	radio	newspaper	sales
230.1	37.8	69.2	22.1
44.5	39.3	45.1	10.4
17.2	45.9	69.3	9.3
151.5	41.3	58.5	18.5
180.8	10.8	58.4	12.9

n observations

p predictors

Heart Data

response variable ***Y*** is Yes/No

Age	Sex	ChestPain	RestBP	Chol	Fbs	RestECG	MaxHR	ExAng	Oldpeak	Slope	Ca	Thal	AHD
63	1	typical	145	233	1	2	150	0	2.3	3	0.0	fixed	No
67	1	asymptomatic	160	286	0	2	108	1	1.5	2	3.0	normal	Yes
67	1	asymptomatic	120	229	0	2	129	1	2.6	2	2.0	reversable	Yes
37	1	nonanginal	130	250	0	0	187	0	3.5	3	0.0	normal	No
41	0	nontypical	130	204	0	2	172	0	1.4	1	0.0	normal	No

Heart Data

These data contain a binary outcome HD for 303 patients who presented with chest pain. An outcome value of:

- ***Yes*** indicates the presence of heart disease based on an angiographic test,
- ***No*** means no heart disease.

There are 13 predictors including:

- Age
- Sex (0 for women, 1 for men)
- Chol (a cholesterol measurement),
- MaxHR
- RestBP

and other heart and lung function measurements.

Classification

Classification

Up to this point, the methods we have seen have centered around modeling and the prediction of a **quantitative** response variable (ex, number of taxi pickups, number of bike rentals, etc). Linear **regression** (and Ridge, LASSO, etc) perform well under these situations

When the response variable is **categorical**, then the problem is no longer called a regression problem but is instead labeled as a **classification problem**.

The goal is to attempt to classify each observation into a category (aka, class or cluster) defined by *Y*, based on a set of predictor variables *X*.

Typical Classification Examples

The motivating examples for this lecture(s), homework, and coming labs are based [mostly] on medical data sets. Classification problems are common in this domain:

- Trying to determine where to set the *cut-off* for some diagnostic test (pregnancy tests, prostate or breast cancer screening tests, etc...)
- Trying to determine if cancer has gone into remission based on treatment and various other indicators
- Trying to classify patients into types or classes of disease based on various genomic markers

Why not Linear Regression?

Simple Classification Example

Given a dataset:

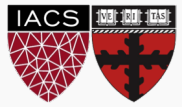
$$\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \cdots, (\boldsymbol{x}_N, y_N)\}$$

where the y are categorical (sometimes referred to as ***qualitative***), we would like to be able to predict which category y takes on given x.

A categorical variable y could be encoded to be quantitative. For example, if y represents concentration of Harvard undergrads, then y could take on the values:

$$y = \begin{cases} 1 & if \text{ Computer Science (CS)} \\ 2 & if \text{ Statistics} \\ 3 & \text{otherwise} \end{cases} .$$

Linear regression **does not work well**, or is not appropriate at all, in this setting.



Simple Classification Example (cont.)

A linear regression could be used to predict y from $\boldsymbol{x}$. What would be wrong with such a model?

The model would imply a specific ordering of the outcome, and would treat a one-unit change in y equivalent. The jump from $y = 1$ to $y = 2$ (CS to Statistics) should not be interpreted as the same as a jump from $y = 2$ to $y = 3$ (Statistics to everyone else).

Similarly, the response variable could be reordered such that $y = 1$ represents Statistics and $y = 2$ represents CS, and then the model estimates and predictions would be fundamentally different.

If the categorical response variable was ***ordinal*** (had a natural ordering, like class year, Freshman, Sophomore, etc.), then a linear regression model would make some sense but is still not ideal.

Even Simpler Classification Problem: Binary Response

The simplest form of classification is when the response variable y has only two categories, and then an ordering of the categories is natural. For example, an upperclassmen Harvard student could be categorized as (note, the $y = 0$ category is a "catch-all" so it would involve both River House students and those who live in other situations: off campus, etc):

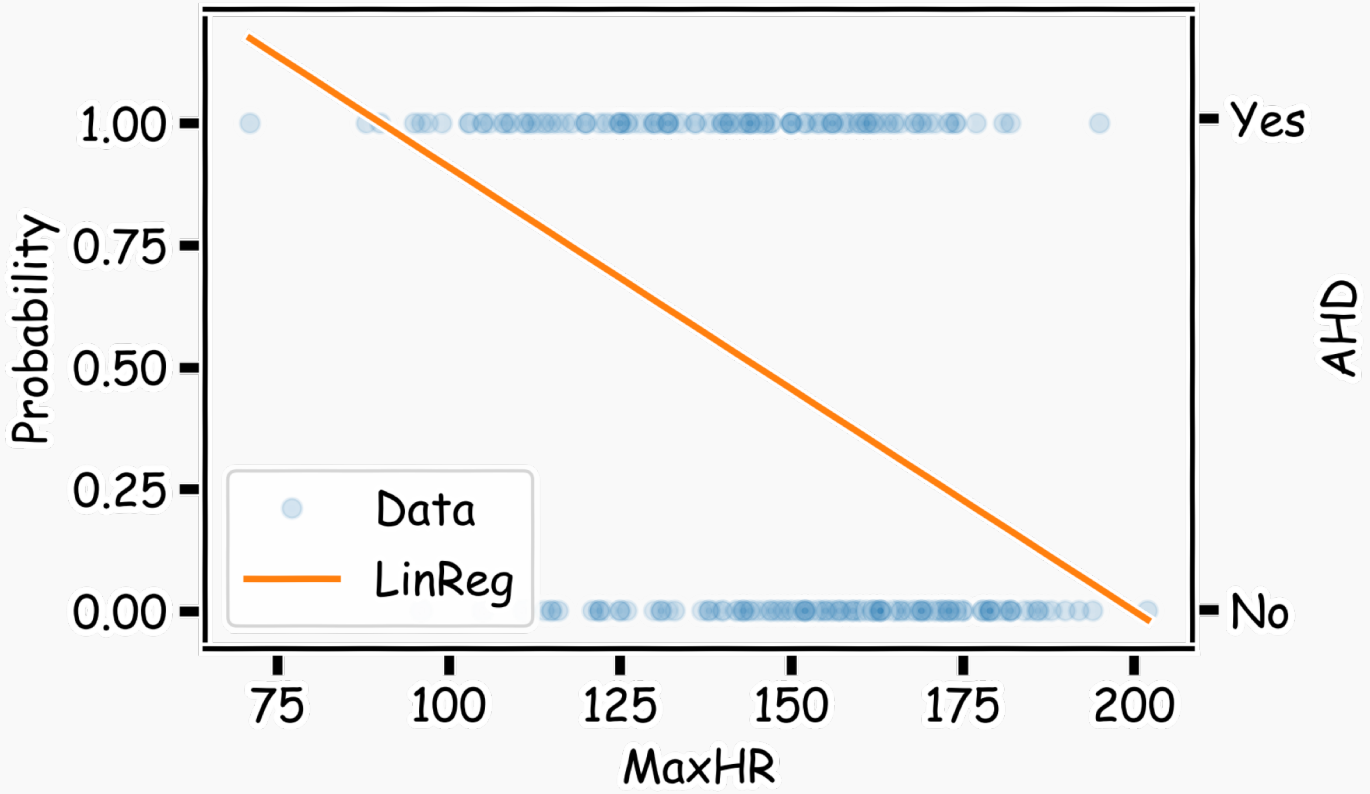
$$y = \begin{cases} 1 & if \text{ lives in the Quad} \\ 0 & \text{otherwise} \end{cases}.$$

Linear regression could be used to predict y directly from a set of covariates (like sex, whether an athlete or not, concentration, GPA, etc.), and if $\hat{y} \geq 0.5$, we could predict the student lives in the Quad and predict other houses if $\hat{y} < 0.5$.

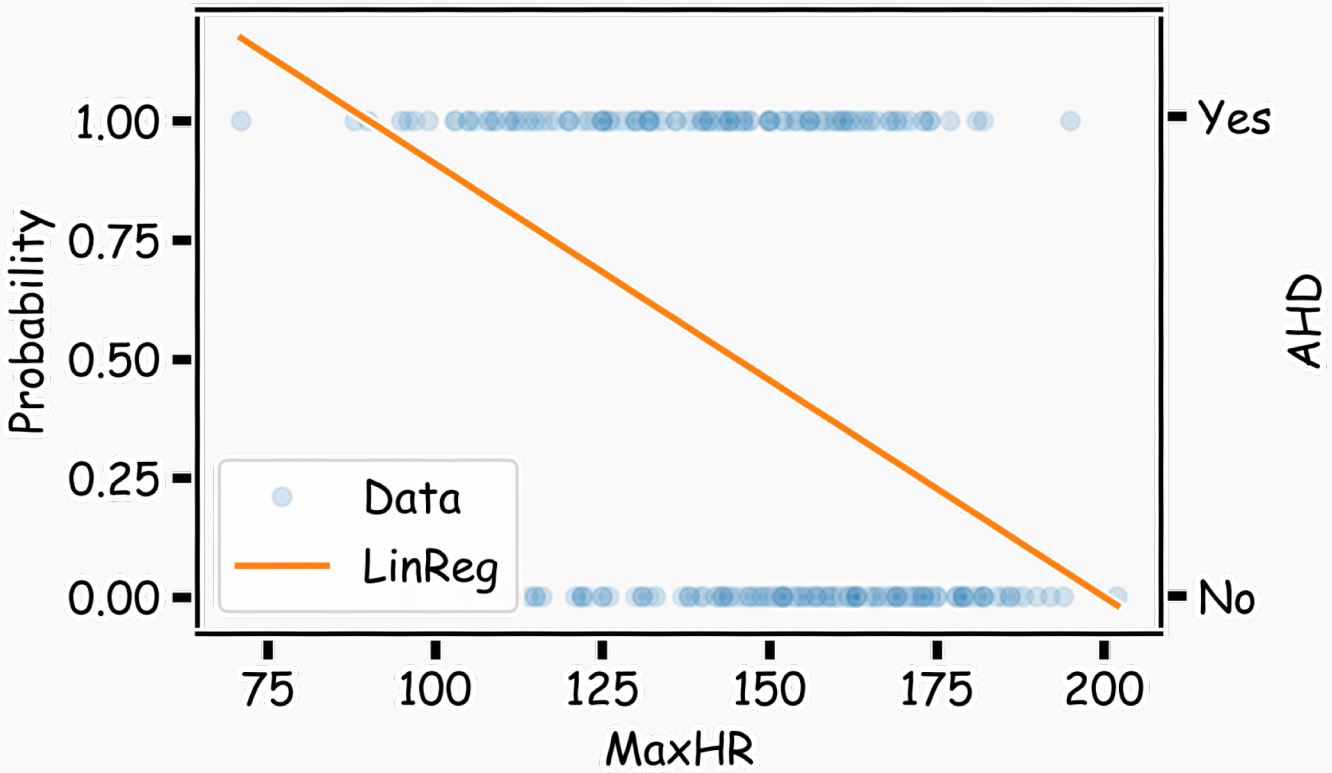
Even Simpler Classification Problem: Binary Response (cont)

What could go wrong with this linear regression model?

.



Even Simpler Classification Problem: Binary Response (cont)



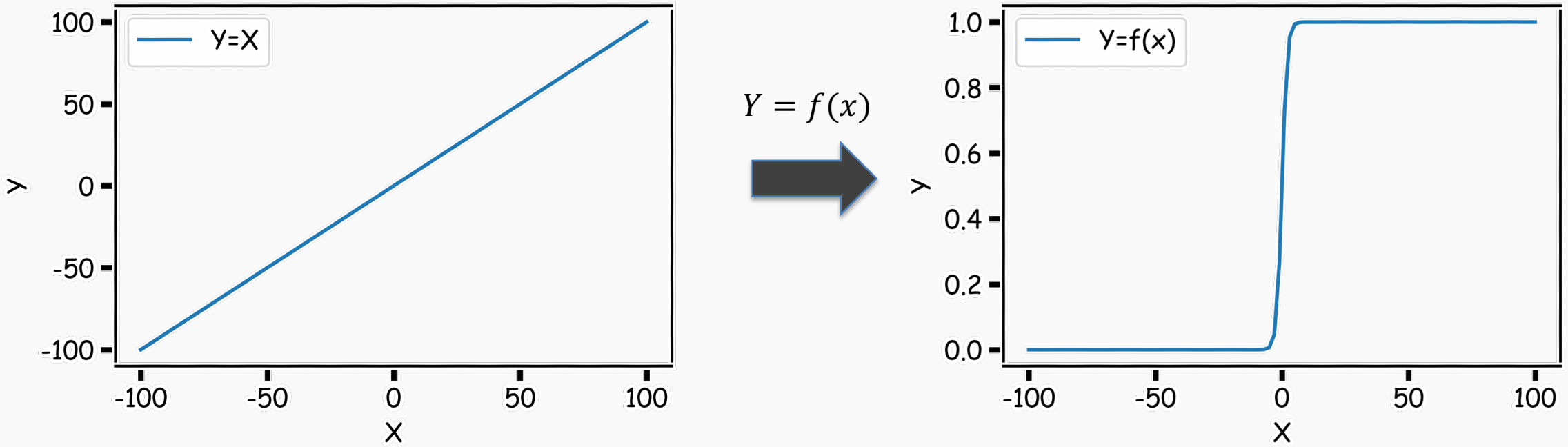
The main issue is you could get non-sensical values for y. Since this is modeling $P(y = 1)$, values for $\hat{y}$ below 0 and above 1 would be at odds with the natural measure for y. Linear regression can lead to this issue.

Lecture Outline

- Classification: Why not Linear Regression?
- **Binary Response & Logistic Regression**
  - **Estimating the Simple Logistic Model**
  - **Classification using the Logistic Model**
  - **Multiple Logistic Regression**
- Classification Boundaries
- Regularization in Logistic Regression
- Bayes Theorem and Misclassification Rates
- ROC Curves
- Multiclass Classification
  - Multinomial and OvR Logistic Regression
  - Multiclass *k*-NN

Pavlos Game #45

Think of a function that would do this for us



Logistic Regression

Logistic Regression addresses the problem of estimating a probability, $P(y = 1)$, to be outside the range of $[0,1]$. The logistic regression model uses a function, called the ***logistic*** function, to model $P(y = 1)$:

$$P(Y = 1) = \frac{e^{\beta_0 + \beta_1 X}}{1 + e^{\beta_0 + \beta_1 X}} = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X)}}$$

Logistic Regression

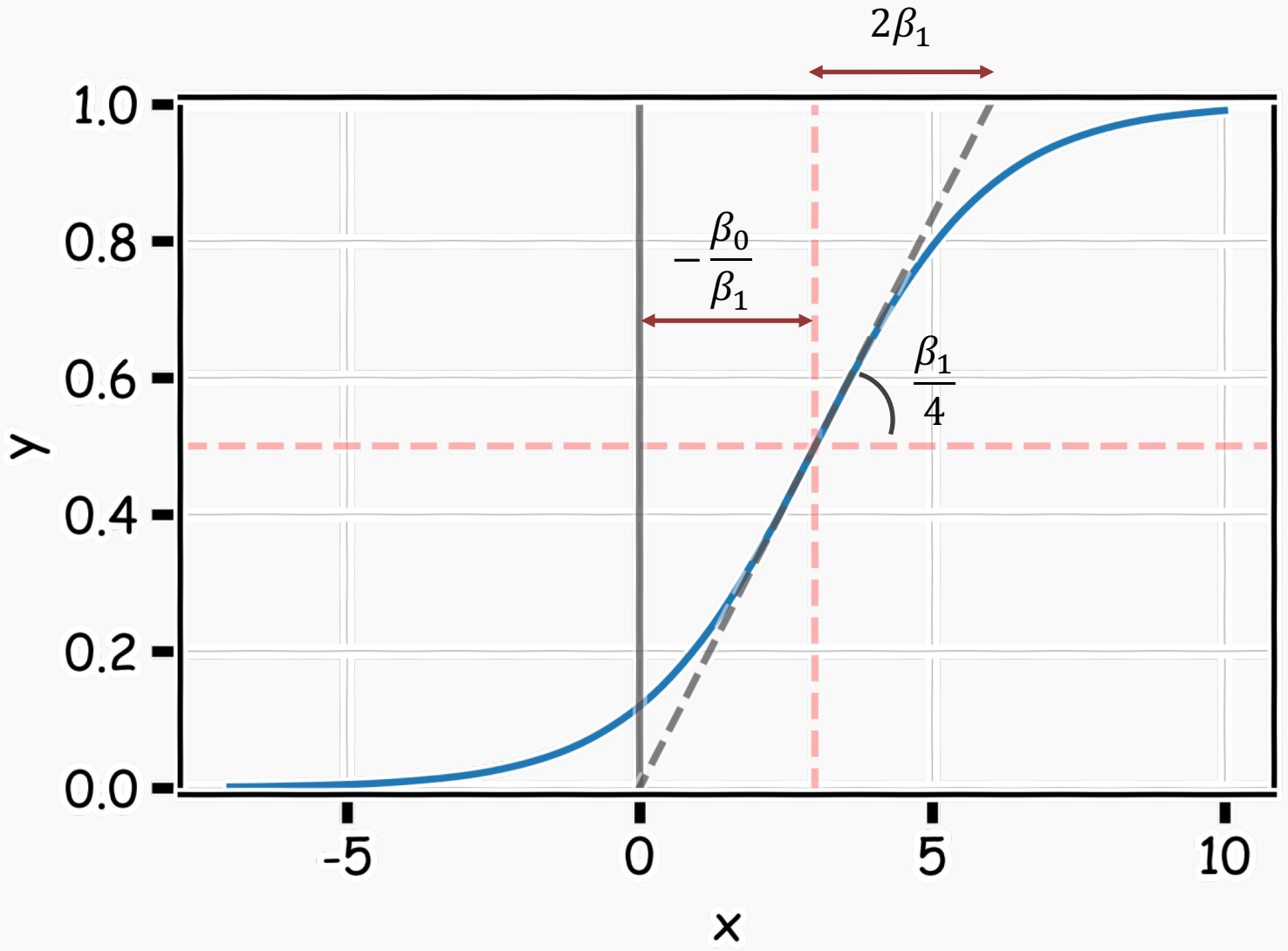
As a result the model will predict $P(y = 1)$ with an S-shaped curve, which is the general shape of the logistic function.

β_0 shifts the curve right or left by $c = -\frac{\beta_0}{\beta_1}$.

β_1 controls how steep the S-shaped curve is. Distance from ½ to almost 1 or ½ to almost 0 to ½ is $\frac{2}{\beta_1}$

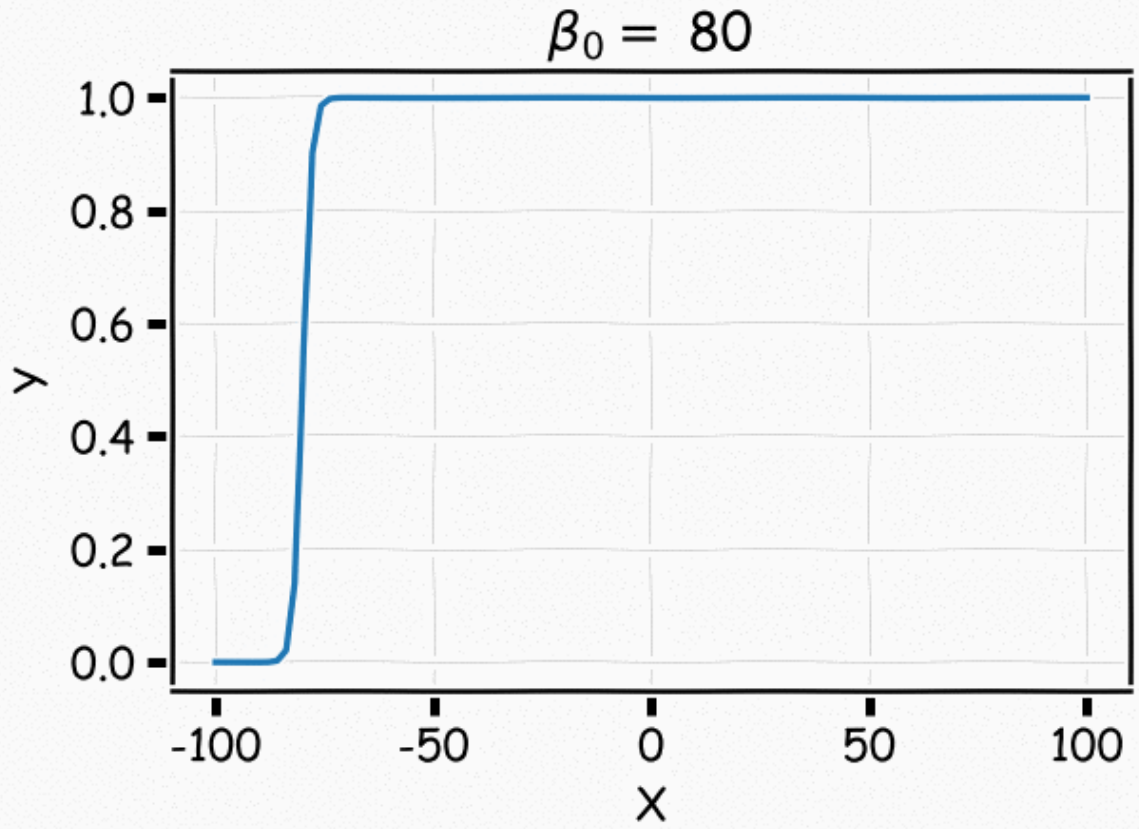
Note: if β_1 is positive, then the predicted $P(y = 1)$ goes from zero for small values of X to one for large values of X and if β_1 is negative, then the $P(y = 1)$ has opposite association.

Logistic Regression



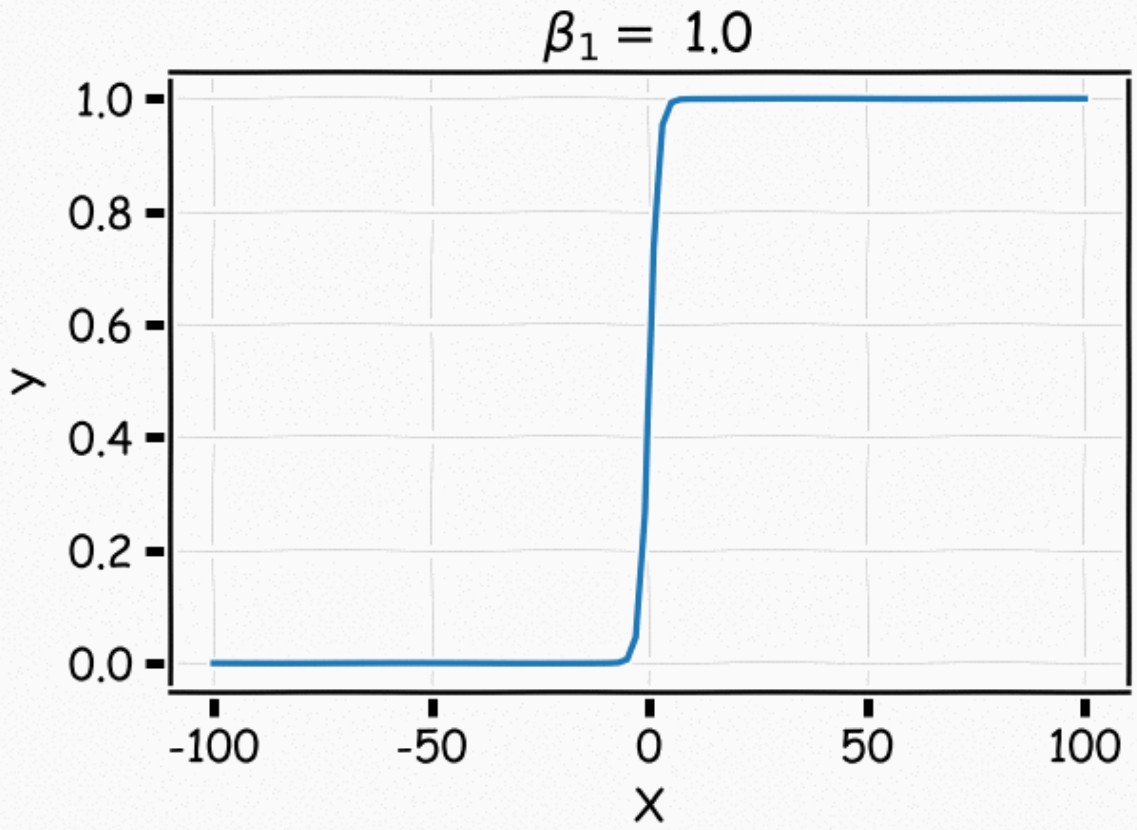
Logistic Regression

$$P(Y = 1) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X)}}$$



Logistic Regression

$$P(Y = 1) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X)}}$$



Logistic Regression

With a little bit of algebraic work, the logistic model can be rewritten as:

$$\ln\left(\frac{P(Y=1)}{1-P(Y=1)}\right) = \beta_0 + \beta_1 X.$$

The value inside the natural log function $\frac{P(Y=1)}{1-P(Y=1)}$, is called the ***odds***, thus logistic regression is said to model the ***log-odds*** with a linear function of the predictors or features, X. This gives us the natural interpretation of the estimates similar to linear regression: a one unit change in X is associated with a β_1 change in the log-odds of $Y = 1$; or better yet, a one unit change in X is associated with an e^{β_1} change in the odds that $Y = 1$.

Estimating the Simple Logistic Model

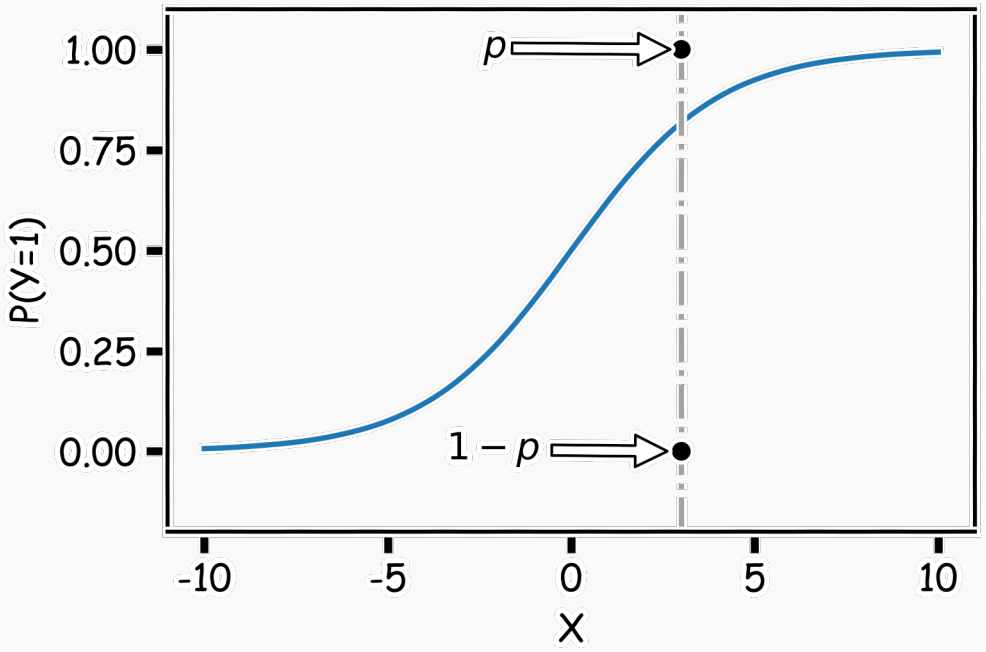
Estimation in Logistic Regression

Unlike in linear regression where there exists a closed-form solution to finding the estimates, $\hat{\beta}_j$'s, for the true parameters, logistic regression estimates cannot be calculated through simple matrix multiplication.

Questions:

- In linear regression what loss function was used to determine the parameter estimates?
- What was the probabilistic perspective on linear regression?
- Logistic Regression also has a likelihood based approach to estimating parameter coefficients.

Estimation in Logistic Regression



Probability $Y = 1$: p

Probability $Y = 0$: $1 - p$

$$P(Y = y) = p^y (1 - p)^{(1-y)}$$

where:

$p = P(Y = 1|X = x)$ and therefore p depends on X.

Thus not every p is the same for each individual measurement.

Likelihood

The likelihood of a single observation for *p* given *x and y* is:

$$L(p_i|Y_i) = P(Y_i = y_i) = p_i^{y_i}(1-p_i)^{1-y_i}$$

Given the observations are independent, what is the likelihood function for *p*?

$$L(p|Y) = \prod_i P(Y_i = y_i) = \prod_i p_i^{y_i}(1-p_i)^{1-y_i}$$

$$l(p|Y) = -\log L(p|Y) = -\sum_i y_i \log p_i + (1-y_i)\log(1-p_i)$$

Loss Function

$$l(p|Y) = -\sum_i \left[y_i \log \frac{1}{1+e^{-\beta X_i}} + (1-y_i) \log\left(1 - \frac{1}{1+e^{-\beta X_i}}\right) \right]_i$$

How do we minimize this?

Differentiate, equate to zero and solve for it!

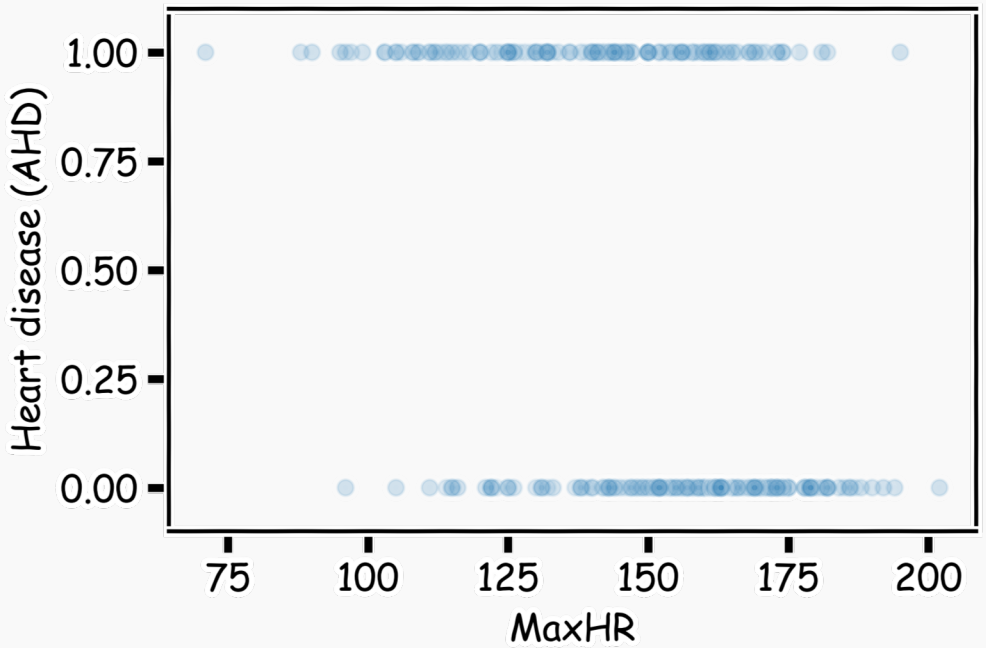
But jeeze does this look messy?! It will not necessarily have a closed form solution.

So how do we determine the parameter estimates? Through an iterative approach (we will talk about this *at length* in future lectures).

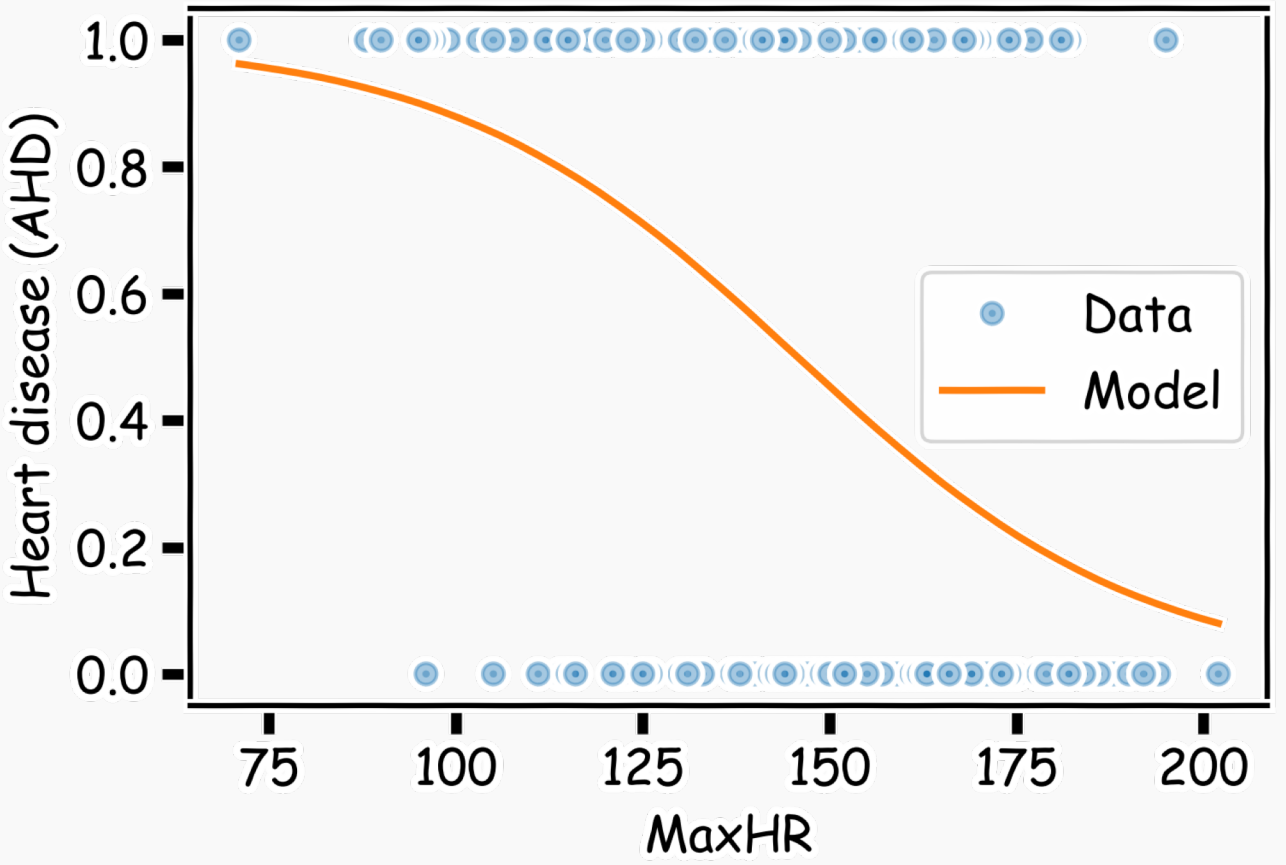
Heart Data: logistic estimation

We'd like to predict whether or not a person has a heart disease. And we'd like to make this prediction, for now, just based on the MaxHR.

How should we visualize these data?



Heart Data: logistic estimation



Heart Data: logistic estimation

There are various ways to fit a logistic model to this data set in Python. The most straightforward in `sklearn` is via `linear_model.LogisticRegression`.

```
from sklearn.linear_model import LogisticRegression

logreg = LogisticRegression(C=100000, fit_intercept=True)
logreg.fit(data_x.values.reshape(-1,1), data_y);



print('Estimated beta1: \n', logreg.coef_)
print('Estimated beta0: \n', logreg.intercept_)
```

```
Estimated beta1:
 [[-0.04326016]]
Estimated beta0:
 [ 6.30193148]
```

Heart Data: logistic estimation

Answer some questions:

- Write down the logistic regression model.
- Interpret $\hat{\beta}_1$.
- Estimate the probability of heart decease for someone (like Pavlos) with MaxHR$\approx$200?
- If we were to use this model purely for classification, how would we do so? See any issues?

Categorical Predictors

Just like in linear regression, when the predictor, X, is binary, the interpretation of the model simplifies (and there is a quick closed form solution).

In this case, what are the interpretations of $\hat{\beta}_0$ and $\hat{\beta}_1$?

For the heart data, let X be the indicator that the individual is a male or female. What is the interpretation of the coefficient estimates in this case?

The observed percentage of HD for women is 26% while it is 55% for men.

Calculate the estimates for $\hat{\beta}_0$ and $\hat{\beta}_1$ if the indicator for HD was predicted from the gender indicator.

Statistical Inference in Logistic Regression

The **uncertainty of the estimates** $\hat{\beta}_0$ and $\hat{\beta}_1$ can be quantified and used to calculate both confidence intervals and hypothesis tests.

The estimate for the standard errors of these estimates, likelihood-based, is based on a quantity called Fisher's Information (beyond the scope of this class), which is related to the curvature of the log-likelihood function.

Due to the nature of the underlying Bernoulli distribution, if you estimate the underlying proportion p_i, you get the variance for free! Because of this, the inferences will be based on the normal approximation (and not t-distribution based). Use statsmodels to perform inferences!

Of course, you could always **bootstrap** the results to perform these inferences as well.

Classification using the Logistic Model

Using Logistic Regression for Classification

How can we use a logistic regression model to perform classification?

That is, how can we predict when $Y = 1$ vs. when $Y = 0$?

We mentioned before, we can classify all observations for which $\hat{P}(Y = 1) \geq 0.5$ to be in the group associated with $Y = 1$ and then classify all observations for which $\hat{P}(Y = 0) < 0.5$ to be in the group associated with $Y = 0$.

Using such an approach is called the standard **Bayes classifier**.

The Bayes classifier takes the approach that assigns each observation to the most likely class, given its predictor values.

Using Logistic Regression for Classification

When will this Bayes classifier be a good one? When will it be a poor one?

The Bayes classifier is the one that minimizes the overall classification error rate. That is, it minimizes:

$$\frac{1}{n}\sum_{i}^{n} I\left(y_i = \hat{y}_i\right)$$

Is this a good Loss function to minimize? Why or why not?

The Bayes classifier may be a poor indicator within a group. Think about the Heart Data scatter plot...

Using Logistic Regression for Classification

This has potential to be a good classifier if the predicted probabilities are on both sides of 0 and 1.

How do we extend this classifier if Y has more than two categories?

Multiple Logistic Regression

Multiple Logistic Regression

It is simple to illustrate examples in logistic regression when there is just one predictors variable.

But the approach ‘easily’ generalizes to the situation where there are multiple predictors.

A lot of the same details as linear regression apply to logistic regression. Interactions can be considered. Multicollinearity is a concern. So is overfitting. Etc...

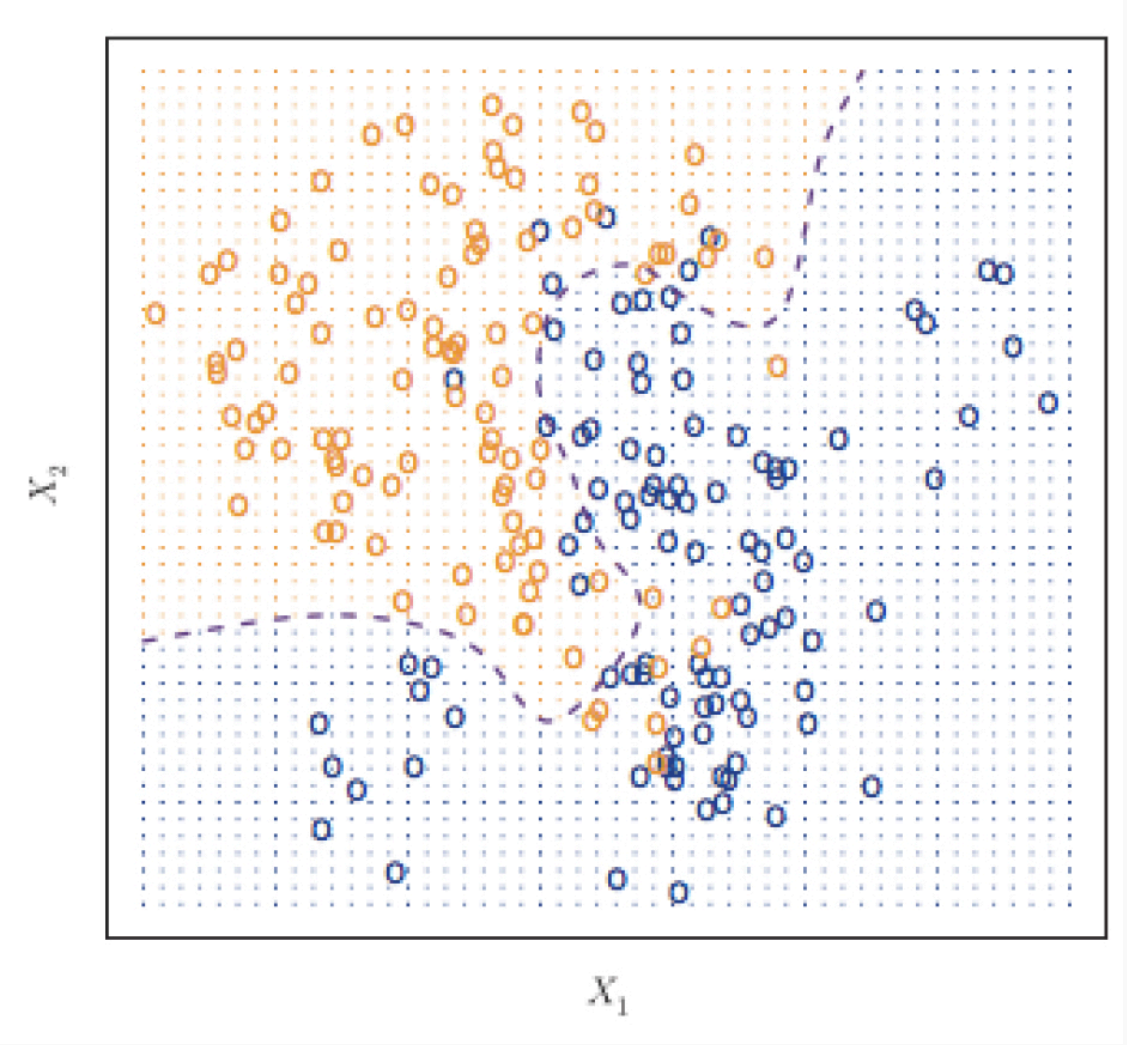
So how do we correct for such problems?

Regularization and checking though train, test, and cross-validation!

We will get into the details of this, along with other extensions of logistic regression, in the next lecture.

Classifier with two predictors

How can we estimate a classifier, based on logistic regression, for the following plot?



Multiple Logistic Regression

Earlier we saw the general form of *simple* logistic regression, meaning when there is just one predictor used in the model. What was the model statement (in terms of linear predictors)?

$$\log\left(\frac{P(Y=1)}{1-P(Y=1)}\right) = \beta_0 + \beta_1 X$$

Multiple logistic regression is a generalization to multiple predictors. More specifically we can define a multiple logistic regression model to predict $P(Y=1)$ as such:

$$\log\left(\frac{P(Y=1)}{1-P(Y=1)}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p$$

Fitting Multiple Logistic Regression

The estimation procedure is identical to that as before for simple logistic regression:

- a likelihood approach is taken, and the function is maximized across all parameters $\beta_0, \beta_1, \ldots, \beta_p$ using an iterative method like Newton-Raphson or Gradient Descent.

The actual fitting of a Multiple Logistic Regression is easy using software (of course there's a python package for that) as the iterative maximization of the likelihood has already been hard coded.

In the `sklearn.linear_model` package, you just have to create your multidimensional design matrix X to be used as predictors in the `LogisticRegression` function.

Interpretation of Multiple Logistic Regression

Interpreting the coefficients in a multiple logistic regression is similar to that of linear regression.

Key: since there are other predictors in the model, the coefficient $\hat{\beta}_j$ is the association between the j^{th} predictor and the response (on log odds scale). But do we have to say? Controlling for the other predictors in the model.

We are trying to attribute the partial effects of each model controlling for the others (aka, controlling for possible *confounders*}.

Interpreting Multiple Logistic Regression: an Example

Let's get back to the Heart Data. We are attempting to predict whether someone has HD based on MaxHR and whether the person is female or male. The simultaneous effect of these two predictors can be brought into one model.

Recall from earlier we had the following estimated models:

$$\log\left(\frac{\widehat{P(Y=1)}}{1-\widehat{P(Y=1)}}\right) = 6.30 - 0.043 \cdot X_{MaxHR}$$

$$\log\left(\frac{\widehat{P(Y=1)}}{1-\widehat{P(Y=1)}}\right) = -1.06 + 1.27 \cdot X_{gender}$$

Interpreting Multiple Logistic Regression: an Example

The results for the multiple logistic regression model are:

```
data_x = df_heart[['MaxHR','Sex']]
data_y = df_heart['AHD']

logreg = LogisticRegression(C=100000, fit_intercept=True)
logreg.fit(data_x, data_y);



print('Estimated beta1: \n', logreg.coef_)
print('Estimated beta0: \n', logreg.intercept_)
```

```
Estimated beta1:
 [[-0.04496354  1.40079047]]
Estimated beta0:
 [ 5.58662464]
```

Some questions

1. Estimate the odds ratio of HD comparing men to women using this model.

2. Is there any evidence of multicollinearity in this model?

3. Is there any confounding in this problem?

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:

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.

Some questions

```
Logistic Regression Estimated Betas (B0,B1,B2,B3): [2.90173899] [[-0.0265903   5.41531302 -0.02707269]]
```

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

3. Estimate the odds ratio of HD comparing men to women using this model [trick question].

Lecture Outline

- Classification: Why not Linear Regression?
- Binary Response & Logistic Regression
  - Estimating the Simple Logistic Model
  - Classification using the Logistic Model
  - Multiple Logistic Regression
- **Classification Boundaries**
- Regularization in Logistic Regression
- Bayes Theorem and Misclassification Rates
- ROC Curves
- Multiclass Classification
  - Multinomial and OvR Logistic Regression
  - Multiclass *k*-NN

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.

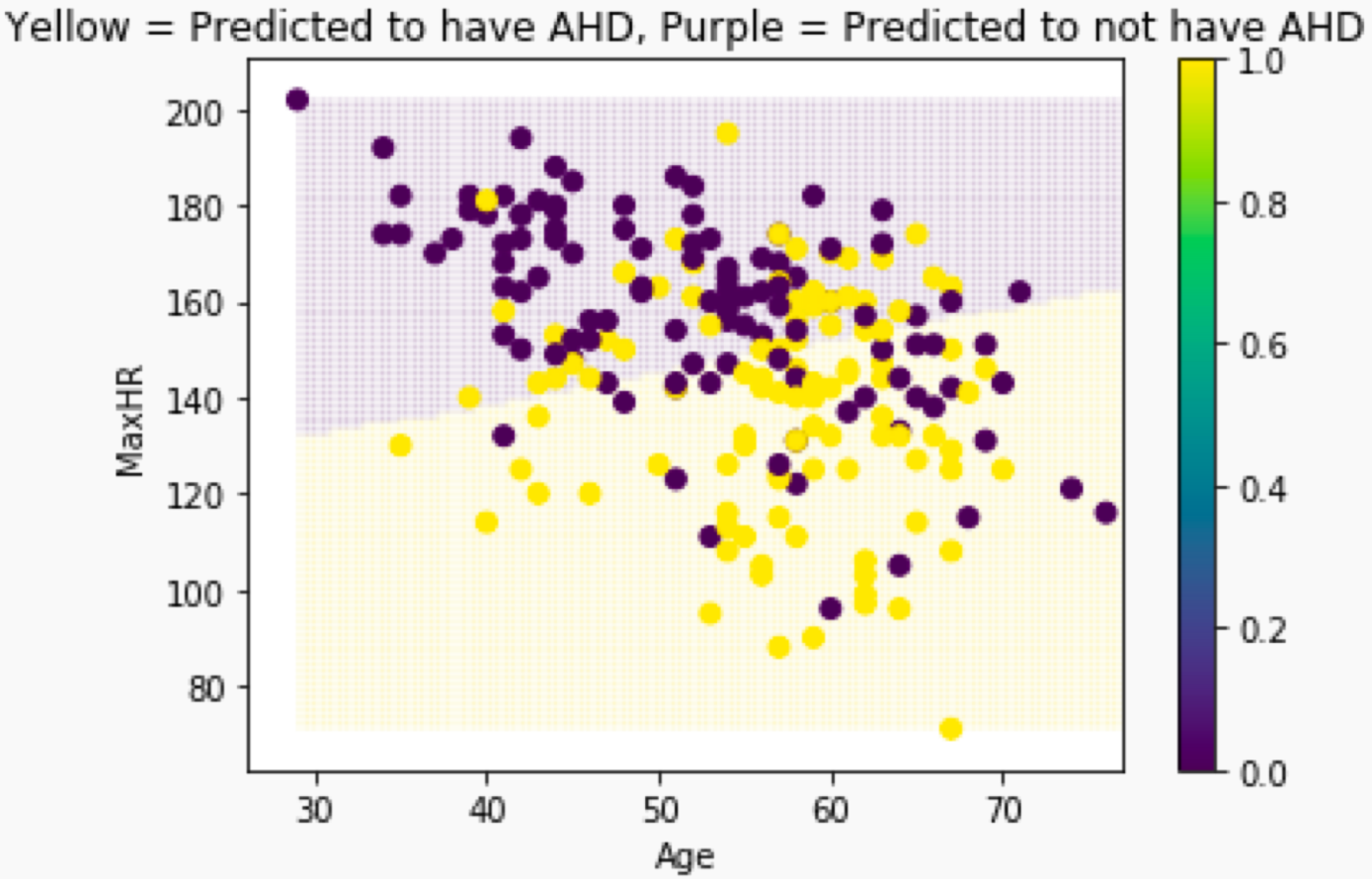
Recall, the logistic regression model statement from last week:

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

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

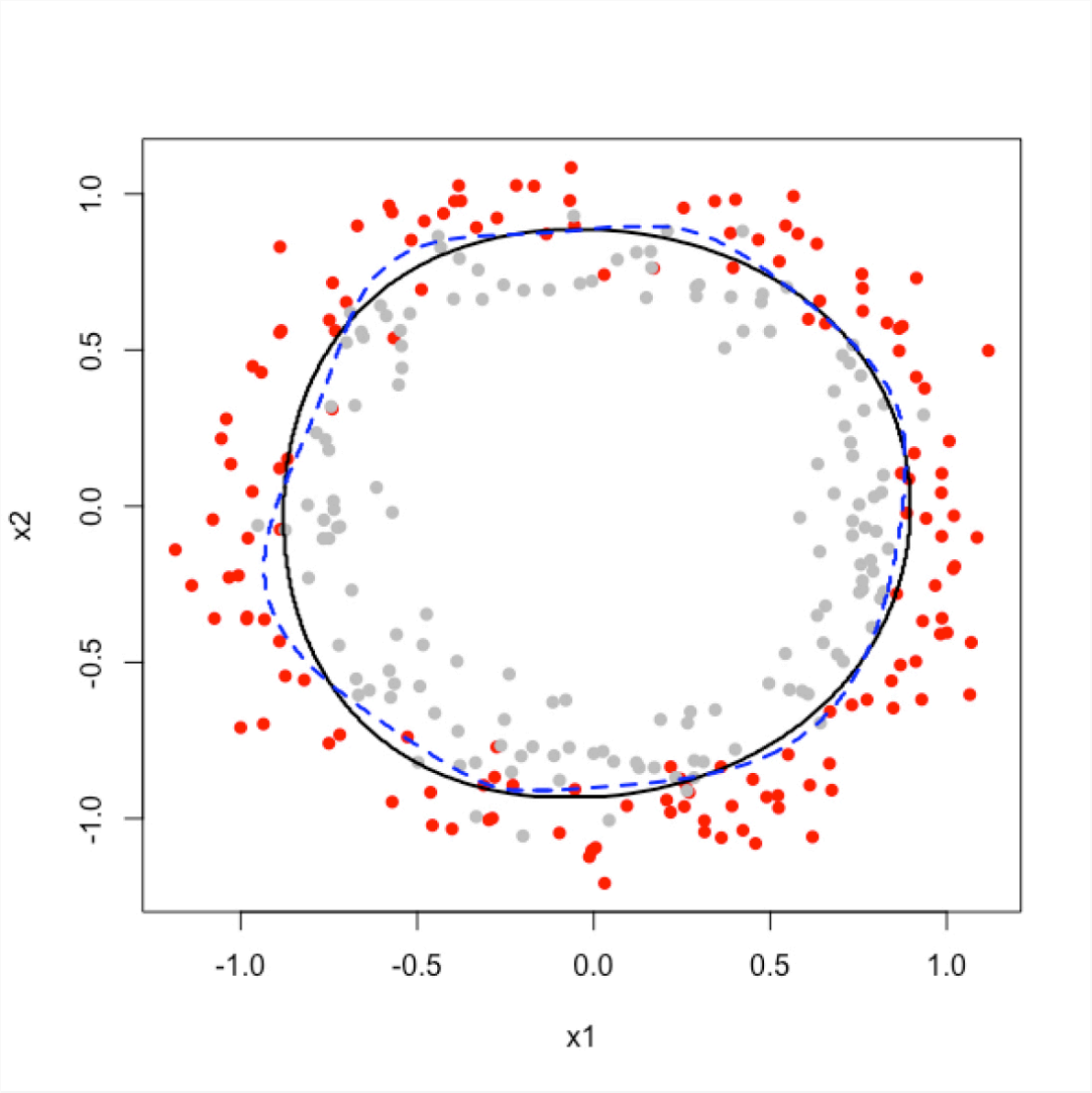


Classification Boundary: a complicated example

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

4. Using the results of the model above, at what ages will males be predicted to have a heart attack in a classification? At what ages will females be predicted to have a heart attack?

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?

Lecture Outline

- Classification: Why not Linear Regression?
- Binary Response & Logistic Regression
  - Estimating the Simple Logistic Model
  - Classification using the Logistic Model
  - Multiple Logistic Regression
- Classification Boundaries
- **Regularization in Logistic Regression**
- Bayes Theorem and Misclassification Rates
- ROC Curves
- Multiclass Classification
  - Multinomial and OvR Logistic Regression
  - Multiclass *k*-NN

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:

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

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.

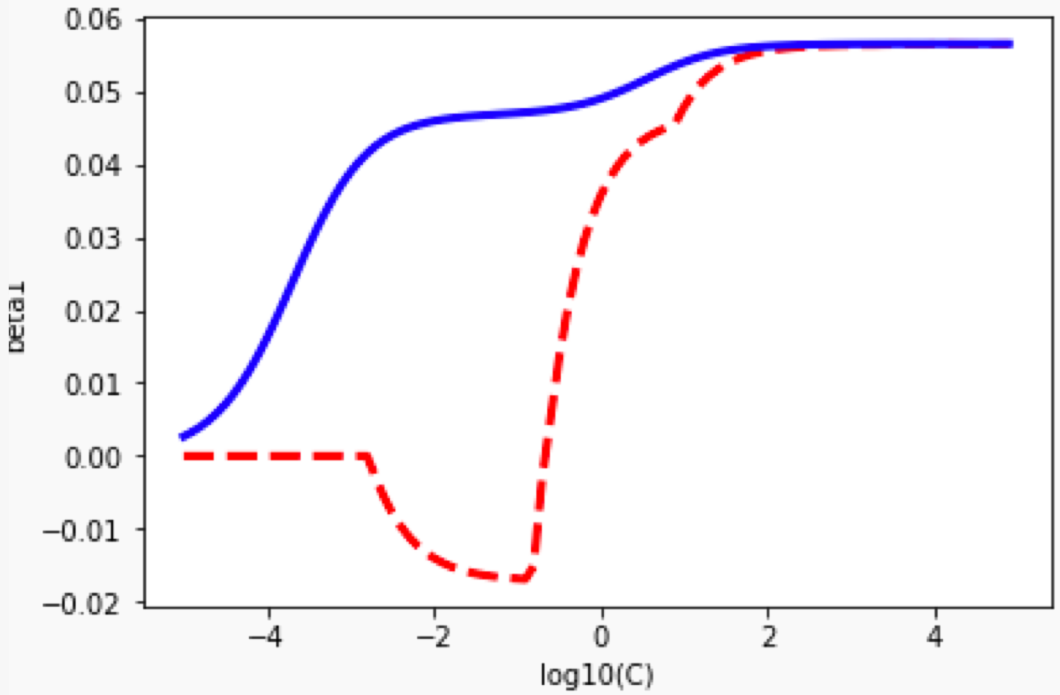
In practice, the intercept is usually not part of the penalty factor.

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 regression.

```
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

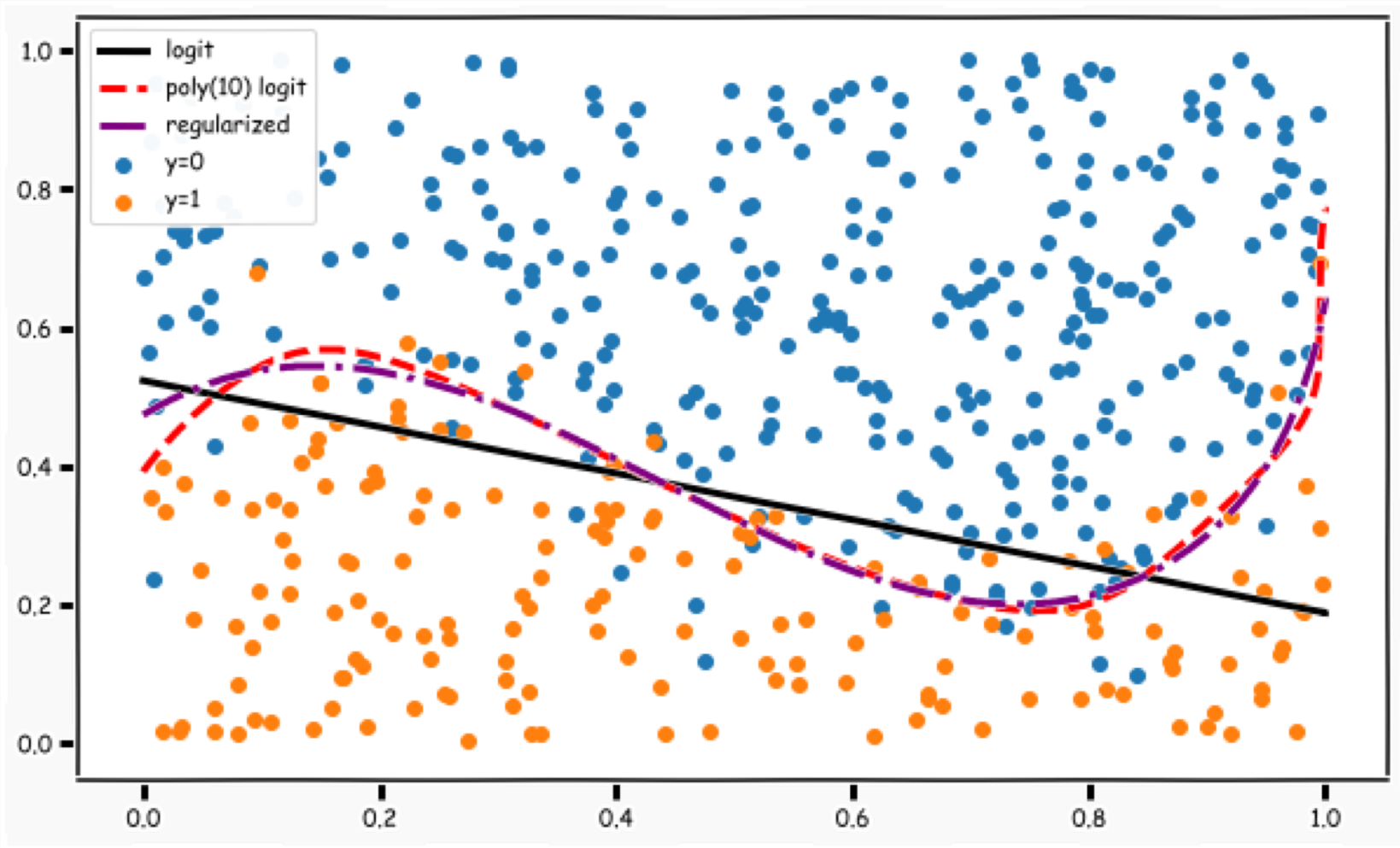
Just like in linear regression, the shrinkage factor must be chosen. How should we go about doing this?

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

How could we measure how well each model fits the validation sets?

We could measure this based on some loss function (more coming later)!

Regularized Decision Boundaries



Lecture Outline

- Classification: Why not Linear Regression?
- Binary Response & Logistic Regression
  - Estimating the Simple Logistic Model
  - Classification using the Logistic Model
  - Multiple Logistic Regression
- Classification Boundaries
- Regularization in Logistic Regression
- **Bayes Theorem and Misclassification Rates**
- ROC Curves
- Multiclass Classification
  - Multinomial and OvR Logistic Regression
  - Multiclass *k*-NN

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. $T-$).

Medical examples of this 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? (hint: it's not just because Kevin is a trained biostatistician!)

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?

Lecture Outline

- Classification: Why not Linear Regression?
- Binary Response & Logistic Regression
  - Estimating the Simple Logistic Model
  - Classification using the Logistic Model
  - Multiple Logistic Regression
- Classification Boundaries
- Regularization in Logistic Regression
- Bayes Theorem and Misclassification Rates
- **ROC Curves**
- Multiclass Classification
  - Multinomial and OvR Logistic Regression
  - Multiclass *k*-NN

ROC Curves

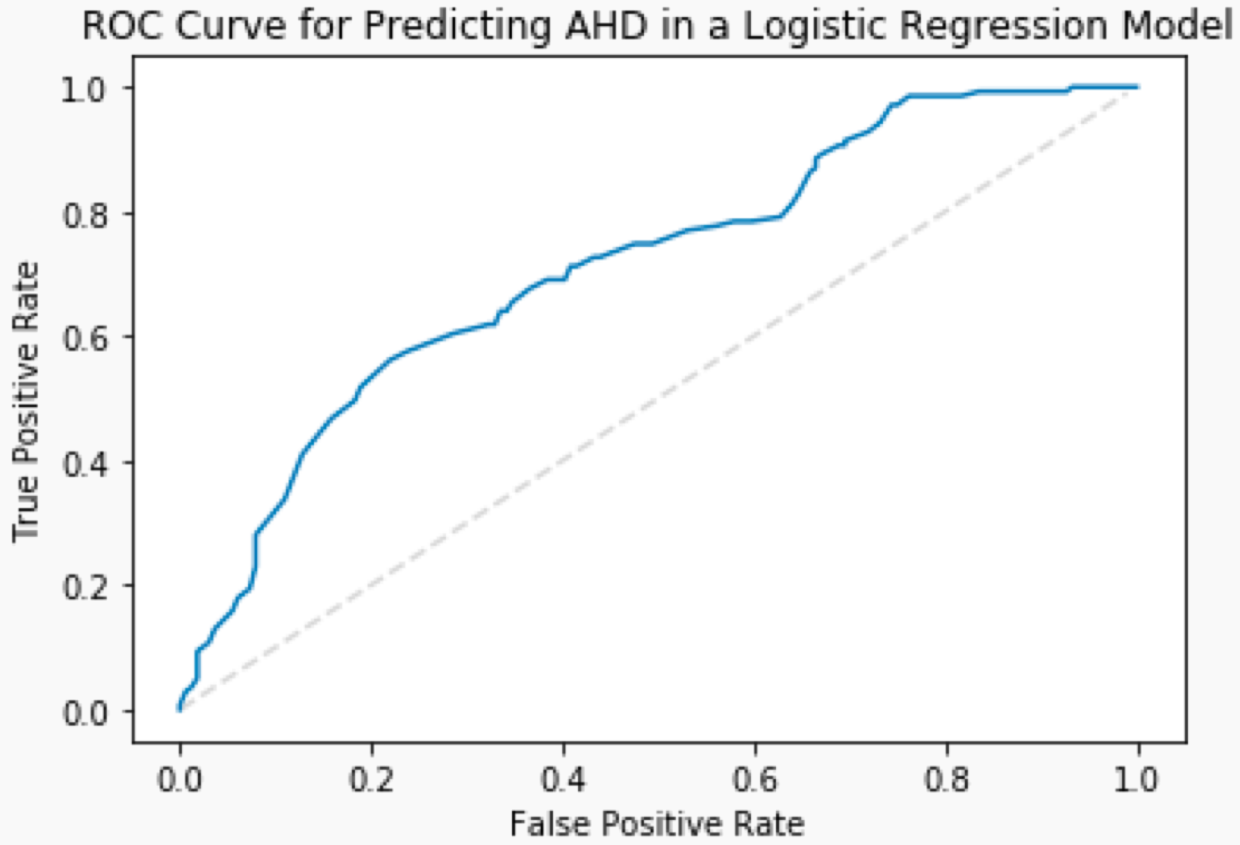
The Radio Operator Characteristics (ROC) curve illustrates the trade-off for all possible thresholds chosen for the two types of error (or correct classification).

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

What is the shape of an ideal ROC curve?

See next slide for an example.

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.

What is the worst case scenario for AUC? What is the best case? What is AUC if we independently just flip a [biased] coin to perform classification?

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

Lecture Outline

- Classification: Why not Linear Regression?
- Binary Response & Logistic Regression
  - Estimating the Simple Logistic Model
  - Classification using the Logistic Model
  - Multiple Logistic Regression
- Classification Boundaries
- Regularization in Logistic Regression
- Bayes Theorem and Misclassification Rates
- ROC Curves
- **Multiclass Classification**
  - **Multinomial and OvR Logistic Regression**
  - **Multiclass *k*-NN**

Logistic Regression for predicting 3+ Classes

There are several extensions to standard logistic regression when the response variable *Y* has more than 2 categories. The two most common are:

1. ordinal logistic regression
2. multinomial logistic regression.

Ordinal logistic regression is used when the categories have a specific hierarchy (like class year: Freshman, Sophomore, Junior, Senior; or a 7-point rating scale from strongly disagree to strongly agree).

Multinomial logistic regression is used when the categories have no inherent order (like eye color: blue, green, brown, hazel, et...).

Multinomial Logistic Regression

Multinomial Logistic Regression

There are two common approaches to estimating a nominal (not-ordinal) categorical variable that has more than 2 classes. The first approach sets one of the categories in the response variable as the *reference* group, and then fits separate logistic regression models to predict the other cases based off of the reference group. For example we could attempt to predict a student's concentration:

$$y = \begin{cases} 1 & if \text{ Computer Science (CS)} \\ 2 & if \text{ Statistics} \\ 3 & \text{otherwise} \end{cases}$$

from predictors X_1 number of psets per week, X_2 how much time playing video games per week, etc.

Multinomial Logistic Regression (cont.)

We could select the $y = 3$ case as the reference group (other concentration), and then fit two separate models: a model to predict $y = 1$ (CS) from $y = 3$ (others) and a separate model to predict $y = 2$ (Stat) from $y = 3$ (others).

Ignoring interactions, how many parameters would need to be estimated?

How could these models be used to estimate the probability of an individual falling in each concentration?

Multinomial Logistic Regression: the model

To predict K classes ($K > 2$) from a set of predictors X, a multinomial logistic regression can be fit:

$$\ln\left(\frac{P(Y=1)}{P(Y=K)}\right) = \beta_{0,1} + \beta_{1,1}X_1 + \beta_{2,1}X_2 + \cdots + \beta_{p,1}X_p$$

$$\ln\left(\frac{P(Y=2)}{P(Y=K)}\right) = \beta_{0,2} + \beta_{1,2}X_1 + \beta_{2,2}X_2 + \cdots + \beta_{p,2}X_p$$

$$\vdots$$

$$\ln\left(\frac{P(Y=K-1)}{P(Y=K)}\right) = \beta_{0,K-1} + \beta_{1,K-1}X_1 + \beta_{2,K-1}X_2 + \cdots + \beta_{p,K-1}X_p$$

Each separate model can be fit as independent standard logistic regression models!

Multinomial Logistic Regression in sklearn

```
mlogit = LogisticRegression(penalty='none',
                            multi_class='multinomial')
mlogit.fit(nfl_19[['ydstogo','down']],nfl_19['playtype'])
print(mlogit.intercept_)
print(mlogit.coef_)
```

```
[-6.78443446  4.46327069  2.32116377]
[[ 0.08130727  1.86026457]
 [-0.10687815 -1.33128781]
 [ 0.02557089 -0.52897676]]
```

```
pd.crosstab(nfl_19["play_type"],
            nfl_19["playtype"])
```

playtype	0	1	2
play_type			
field_goal	978	0	0
pass	0	0	18981
punt	2150	0	0
qb_kneel	393	0	0
qb_spike	72	0	0
run	0	12994	0

But wait Kevin, I thought you said we only fit $K - 1$ logistic regression models!?!? Why are there K intercepts and K sets of coefficients????

What is sklearn doing?

The *K* - 1 models in multinomial regression lead to the following probability predictions:

$$\ln\left(\frac{P(Y=k)}{P(Y=K)}\right) = \beta_{0,k} + \beta_{1,k}X_1 + \beta_{2,k}X_k + \cdots + \beta_{p,k}X_p$$

$$\vdots$$

$$P(Y=k) = P(Y=K)e^{\beta_{0,k}+\beta_{1,k}X_1+\beta_{2,k}X_k+\cdots+\beta_{p,k}X_p}$$

This give us *K* – 1 equations to estimate *K* probabilities for everyone. But probabilities add up to 1 ☺, so we are all set.

Sklearn then converts the above probabilities back into new betas (just like logistic regression, but the betas won't match):

$$\ln\left(\frac{P(Y=k)}{P(Y\neq k)}\right) = \beta'_{0,k} + \beta'_{1,k}X_1 + \beta'_{2,k}X_k + \cdots + \beta'_{p,k}X_p$$

One vs. Rest (OvR) Logistic Regression

One vs. Rest (OvR) Logistic Regression

An alternative multiclass logistic regression model in sklearn is called the 'One vs. Rest' approach, which is our second method.

If there are 3 classes, then 3 separate logistic regressions are fit, where the probability of each category is predicted over the rest of the categories combined. So for the concentration example, 3 models would be fit:

- a first model would be fit to predict CS from (Stat and Others) combined.
- a second model would be fit to predict Stat from (CS and Others) combined.
- a third model would be fit to predict Others from (CS and Stat) combined.

An example to predict play call from the NFL data follows...

One vs. Rest (OvR) Logistic Regression: the model

To predict K classes ($K > 2$) from a set of predictors X, a multinomial logistic regression can be fit:

$$\ln\left(\frac{P(Y=1)}{P(Y\neq 1)}\right) = \beta_{0,1} + \beta_{1,1}X_1 + \beta_{2,1}X_2 + \cdots + \beta_{p,1}X_p$$

$$\ln\left(\frac{P(Y=2)}{P(Y\neq 2)}\right) = \beta_{0,2} + \beta_{1,2}X_1 + \beta_{2,2}X_2 + \cdots + \beta_{p,2}X_p$$

$$\vdots$$

$$\ln\left(\frac{P(Y=K)}{P(Y\neq K)}\right) = \beta_{0,K} + \beta_{1,K}X_1 + \beta_{2,K}X_2 + \cdots + \beta_{p,K}X_p$$

Again, each separate model can be fit as independent standard logistic regression models!

Softmax

So how do we convert a set of probability estimates from separate models to one set of probability estimates?

The ***softmax*** function is used. That is, the weights are just normalized for each predicted probability. AKA, predict the 3 class probabilities from each of the 3 models, and just rescale so they add up to 1.

Mathematically that is:

$$P(y = k|\vec{x}) = \frac{e^{\vec{x}^T \hat{\vec{\beta}}_k}}{\sum_{j=1}^{K} e^{\vec{x}^T \hat{\vec{\beta}}_j}}$$

where $\vec{x}$ is the vector of predictors for that observation and $\hat{\vec{\beta}}_k$ are the associated logistic regression coefficient estimates.

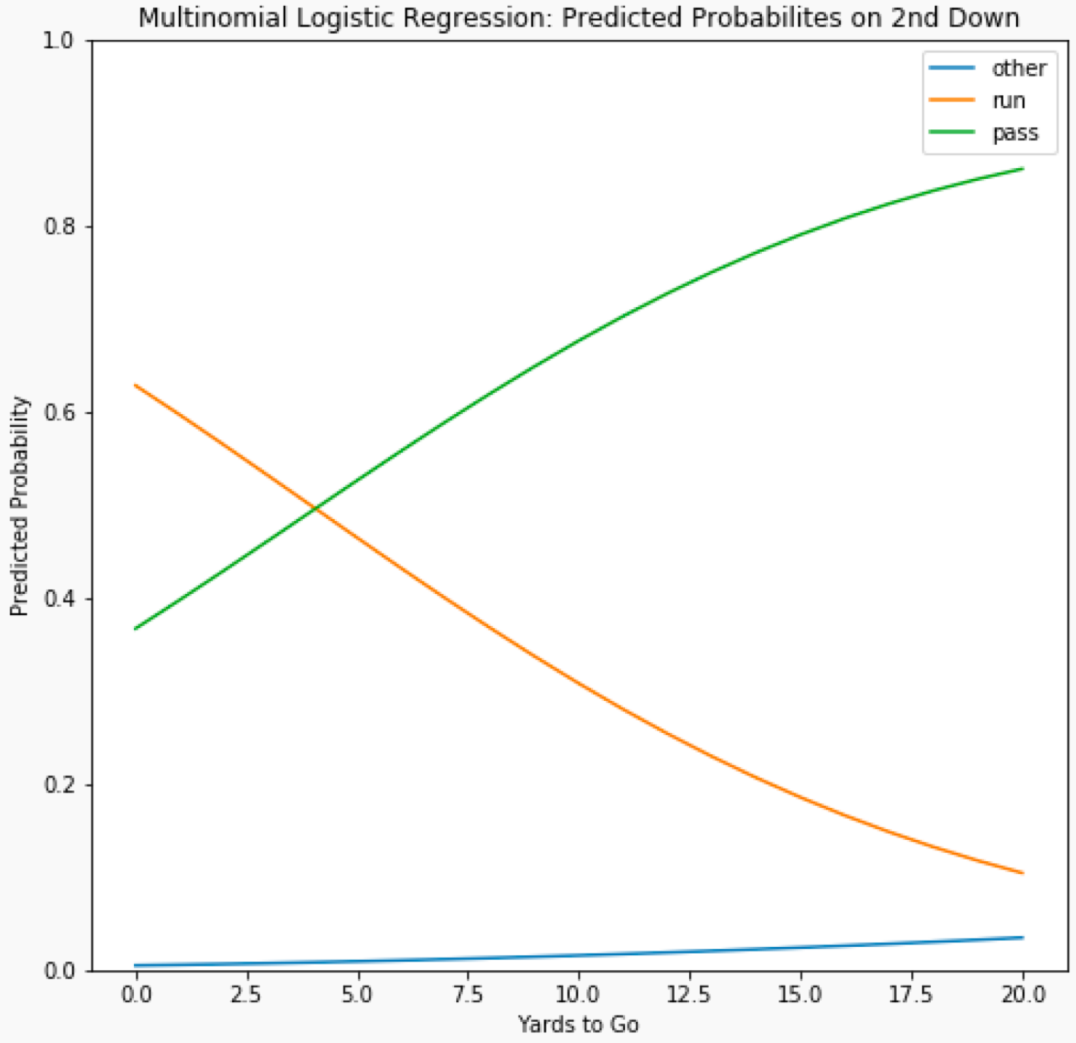
OVR Logistic Regression in Python

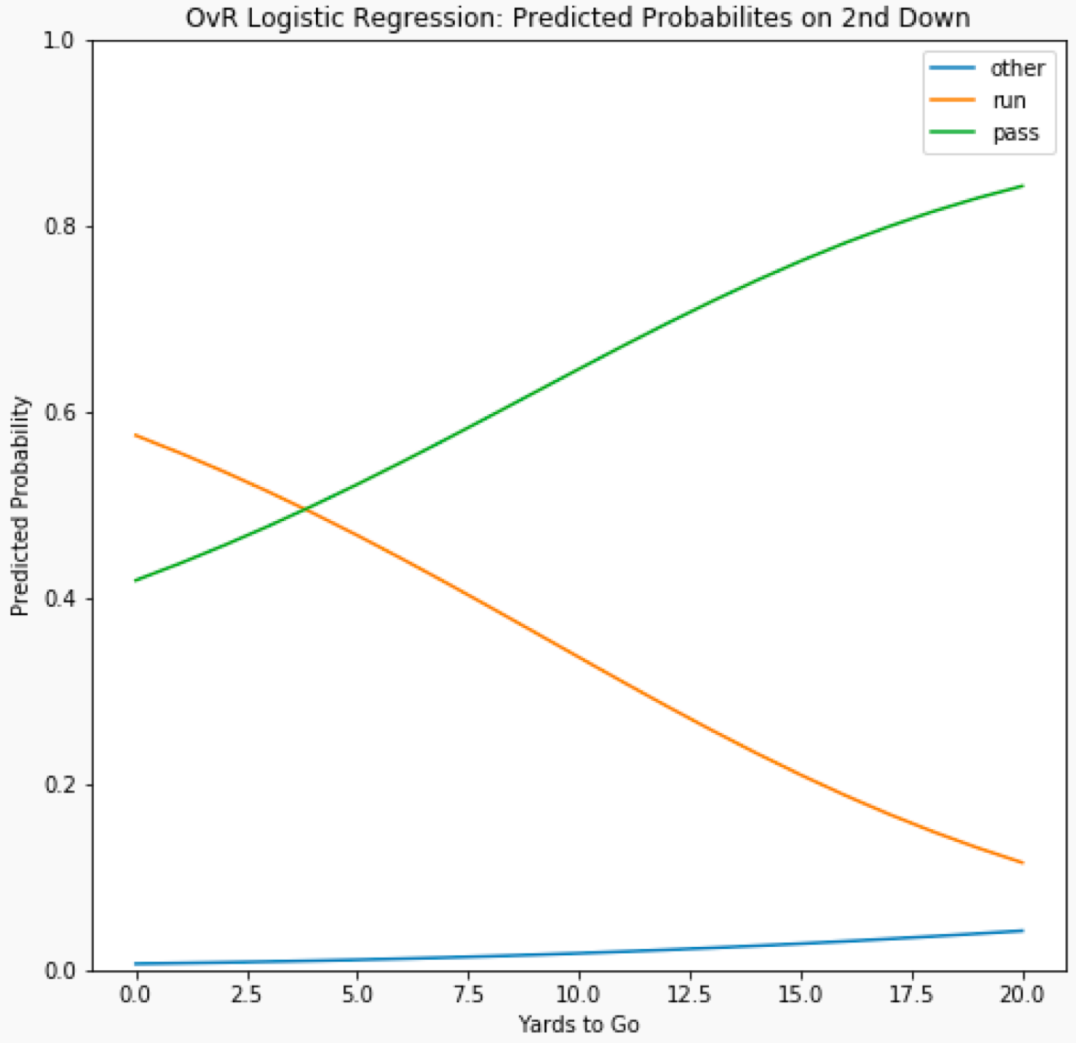
```
ovrlogit = LogisticRegression(penalty='none', multi_class='ovr')

ovrlogit.fit(nfl_19[['ydstogo','down']],nfl_19['playtype'])
print(ovrlogit.intercept_)
print(ovrlogit.coef_)
```

```
[-10.03308293   2.47802369  -0.10976034]
[[ 0.07547391  2.55126265]
 [-0.14272983 -0.98841346]
 [ 0.03672441 -0.03755844]]
```

Phew! This one is as expected ☺

Predicting Type of Play in the NFL





Classification for more than 2 Categories

When there are more than 2 categories in the response variable, then there is no guarantee that $P(Y = k) \geq 0.5$ for any one category. So any classifier based on logistic regression (or other classification model) will instead have to select the group with **the largest estimated probability**.

The classification boundaries are then much more difficult to determine mathematically. We will not get into the algorithm for determining these in this class, but we can rely on `predict` and `predict_proba`!

Prediction using Multiclass Logistic Regression

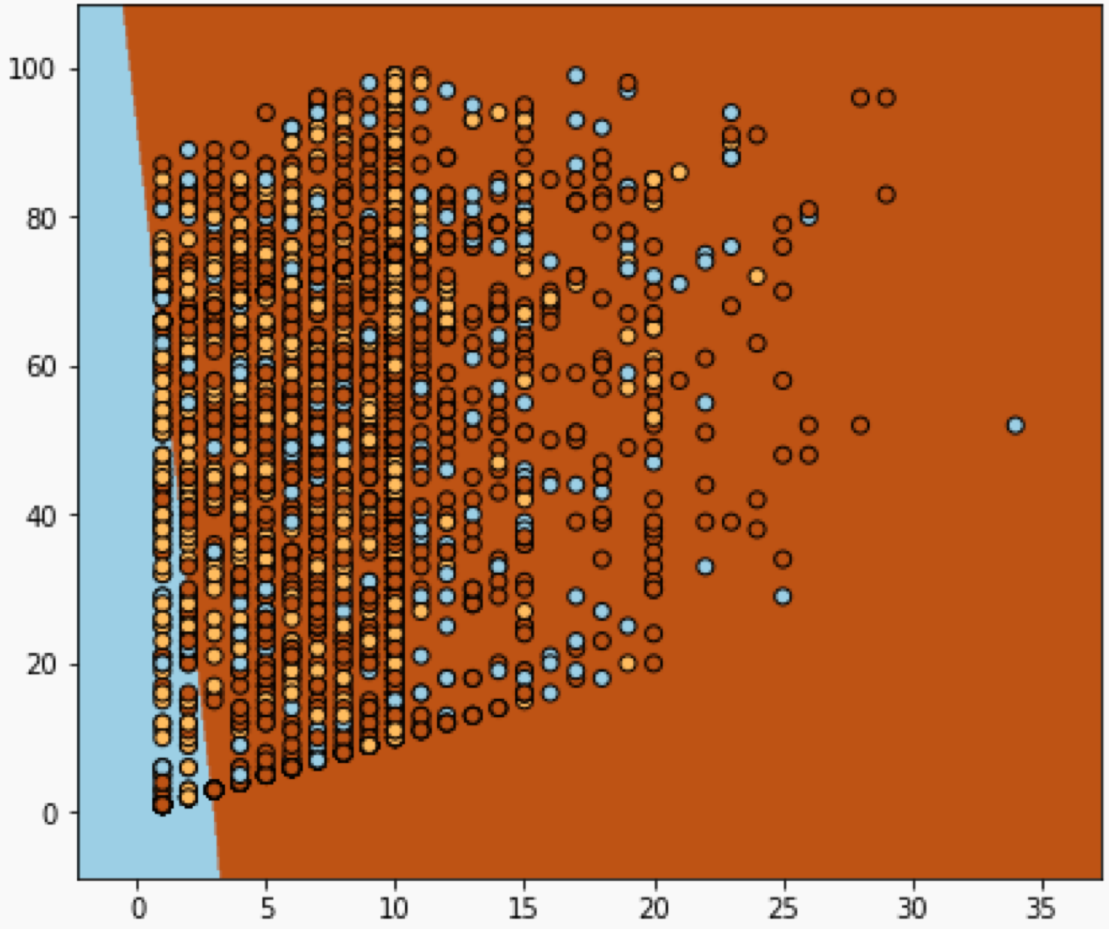
```
print(mlogit.predict_proba(nfl_19[['ydstogo','down']])[0:5,:])
print(mlogit.predict(nfl_19[['ydstogo','down']])[0:5])
```

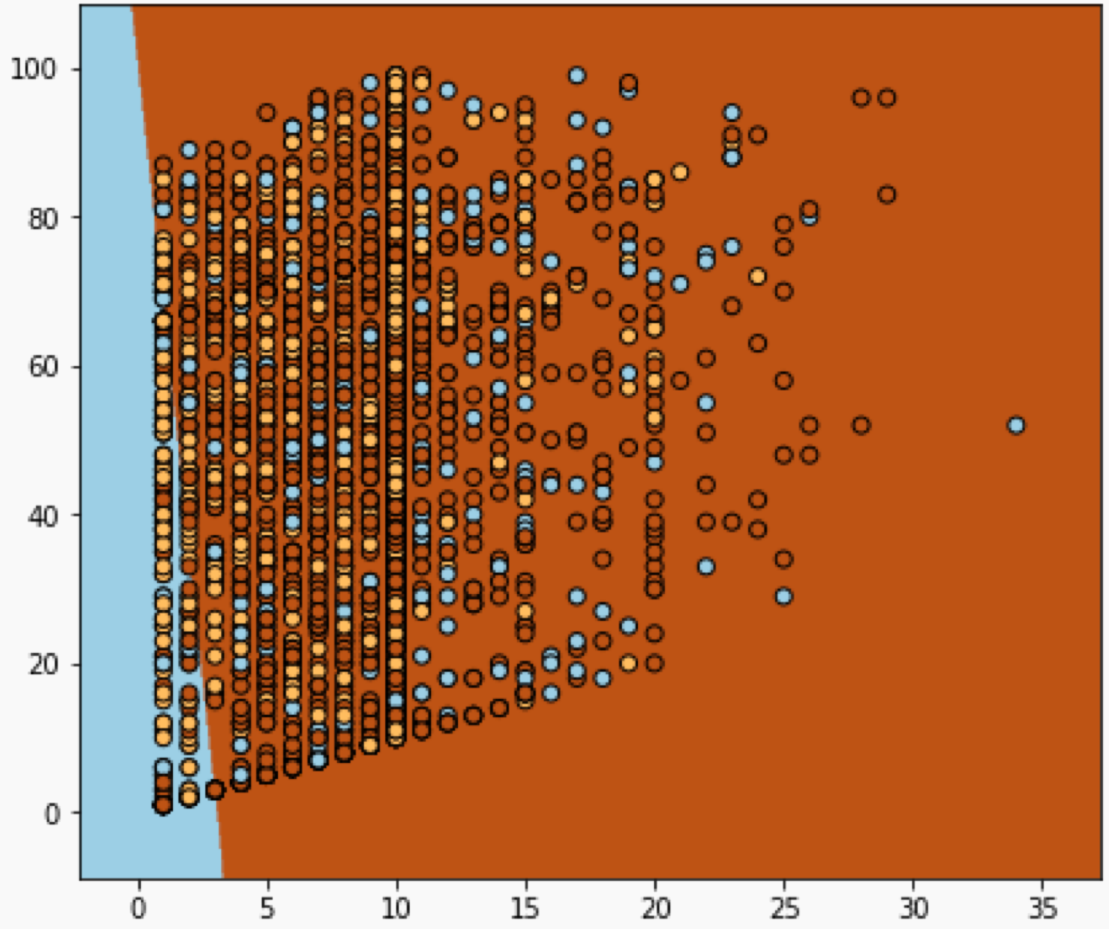
```
[[0.001048   0.50329827 0.49565372]
 [0.01559789 0.30793278 0.67646934]
 [0.17275682 0.14020122 0.68704196]
 [0.82376365 0.00418569 0.17205066]
 [0.001048   0.50329827 0.49565372]]
[1 2 2 0 1]
```

```
print(ovrlogit.predict_proba(nfl_19[['ydstogo','down']])[0:5,:])
print(ovrlogit.predict(nfl_19[['ydstogo','down']])[0:5])
```

```
[[0.00111671 0.48115571 0.51772757]
 [0.01792012 0.33603242 0.64604746]
 [0.19847963 0.15493954 0.64658083]
 [0.57254676 0.0088239  0.41862935]
 [0.00111671 0.48115571 0.51772757]]
[2 2 2 0 2]
```

Classification Boundary for 3+ Classes in sklearn





Estimation and Regularization in multiclass settings

There is no difference in the approach to estimating the coefficients in the multiclass setting: we maximize the log-likelihood (or minimize negative log-likelihood).

This combined negative log-likelihood of all *K* classes is sometimes called the **binary cross-entropy**:

$$\ell = \frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{K} \mathbb{1}(y_i = k)\ln(\hat{P}\,(y_i = k)) + \mathbb{1}(y_i \neq k)\ln(1 - \hat{P}\,(y_i = k))$$

And regularization can be done like always: add on a penalty term to this loss function based on L1 (sum of the absolute values) or L2 (sum of squares) norms.

Multiclass *k*-NN

k-NN for 3+ Classes

Extending the *k*-NN classification model to 3+ classes is much simpler!

Remember, *k*-NN is done in two steps:

1. **Find you *k* neighbors**: this is still done in the exact same way
   - be careful of the scaling of your predictors!
2. **Predict based on your neighborhood:**
   - Predicting probabilities: just use the observed proportions
   - Predicting classes: plurality wins!

k-NN for 3+ Classes: NFL Data