Announcements

Discussion tomorrow will be exam review.

- 9 AM to 3 PM.
- Will focus on going through/solve old exam problems.

Free attendance point today.

DS100: Fall 2019

Lecture 22 (Josh Hug): Classification II

- Decision Boundaries
- More Classifier Evaluation
- Regularization
- Gradient Descent Demo
- Multiclass Classification

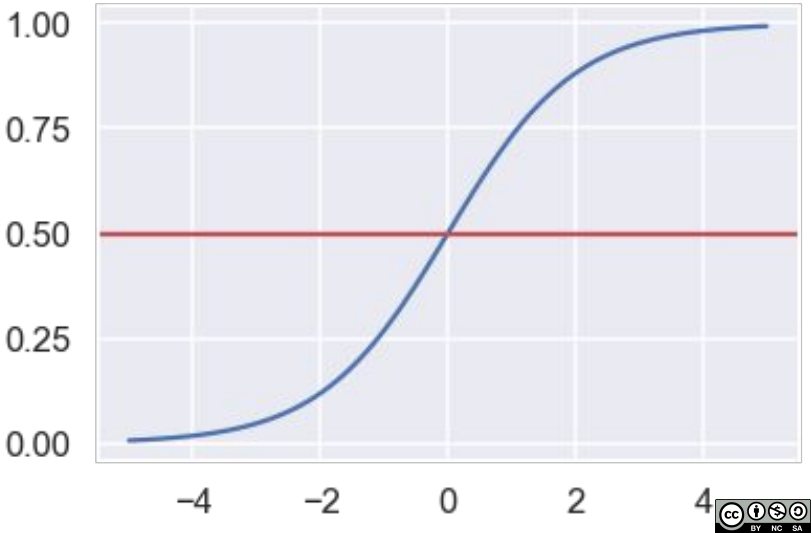
Logistic Regression and Classification Summary

Key ideas:

- A **classifier** is a function that outputs a prediction of **0 or 1** for y.
- **Logistic regression** finds a function estimates **P(Y=1|X)**.
- Need a **decision rule** (aka **classification rule**) to convert from probability to class.

Example for threshold of 0.5:

$$\text{classify}(\vec{x}) = \begin{cases} 1 & P(Y=1|X=\vec{x}) \geq \frac{1}{2} \\ 0 & P(Y=1|X=\vec{x}) < \frac{1}{2} \end{cases}$$

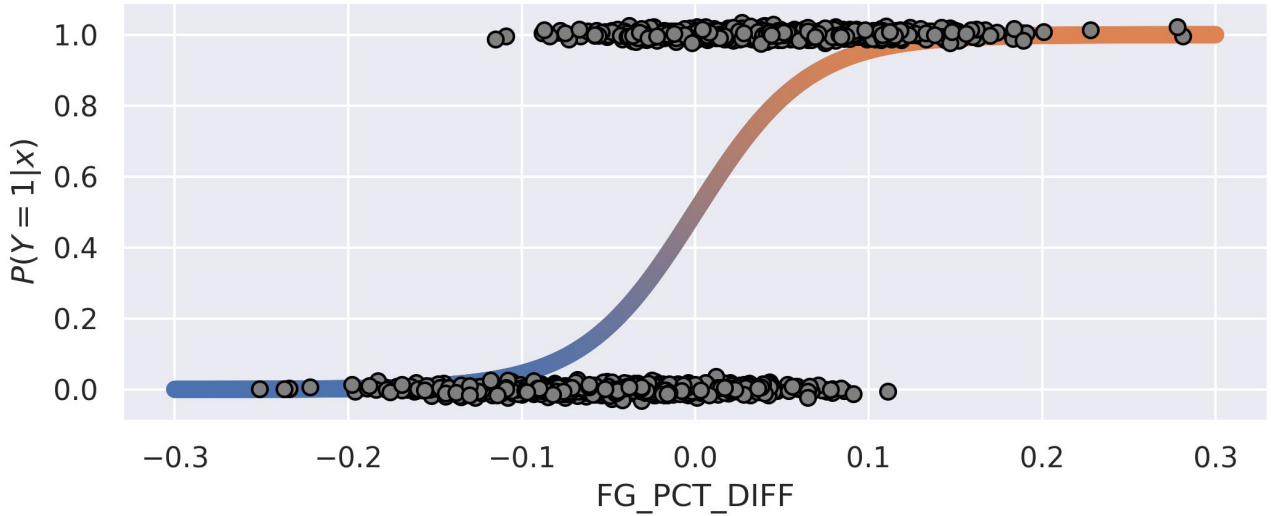


Decision Boundaries

Our Model

Our regression models produce probabilities.

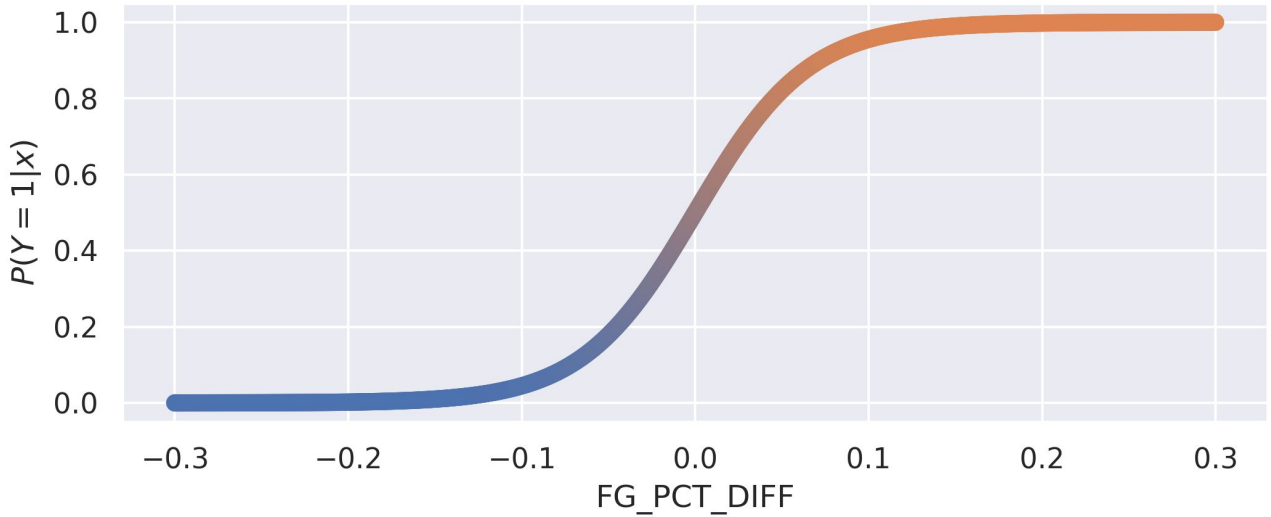
- The curve below is the probability that our model thinks a team will win based on FG_PCT_DIFF (described in an earlier lecture).
- Gray dots are the data from the 2018-2019 NBA season.



Our Model

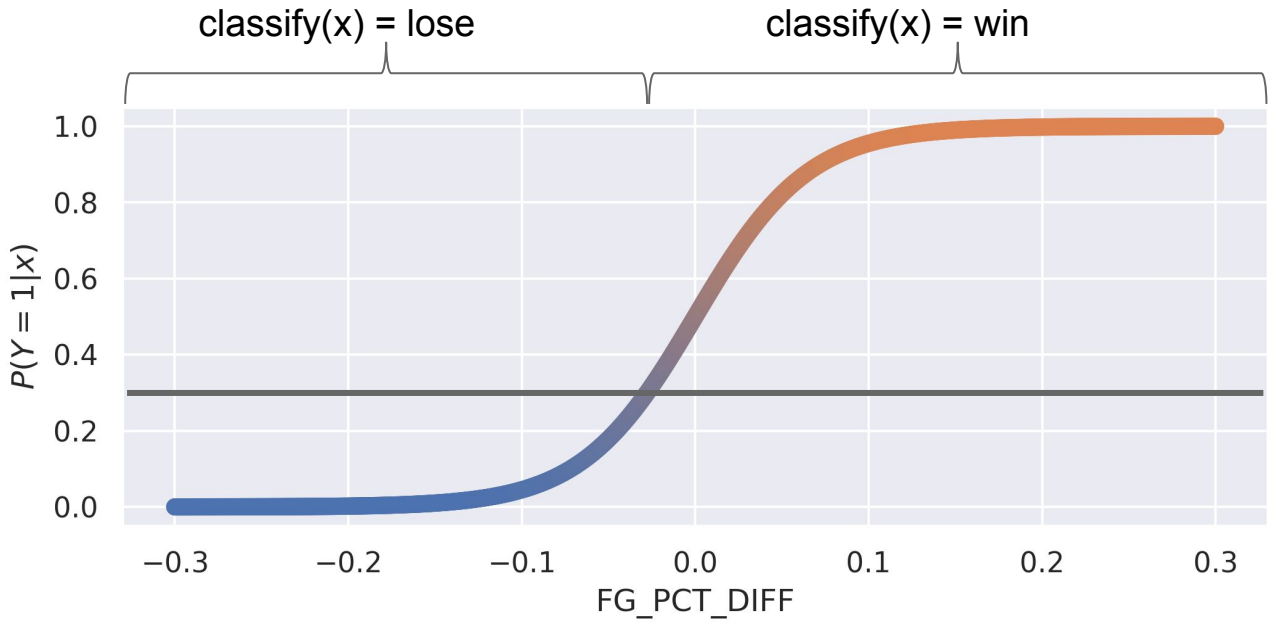
Our regression models produce probabilities.

- Let's ignore the data (gray dots) and focus on just the predictions.



Our Model

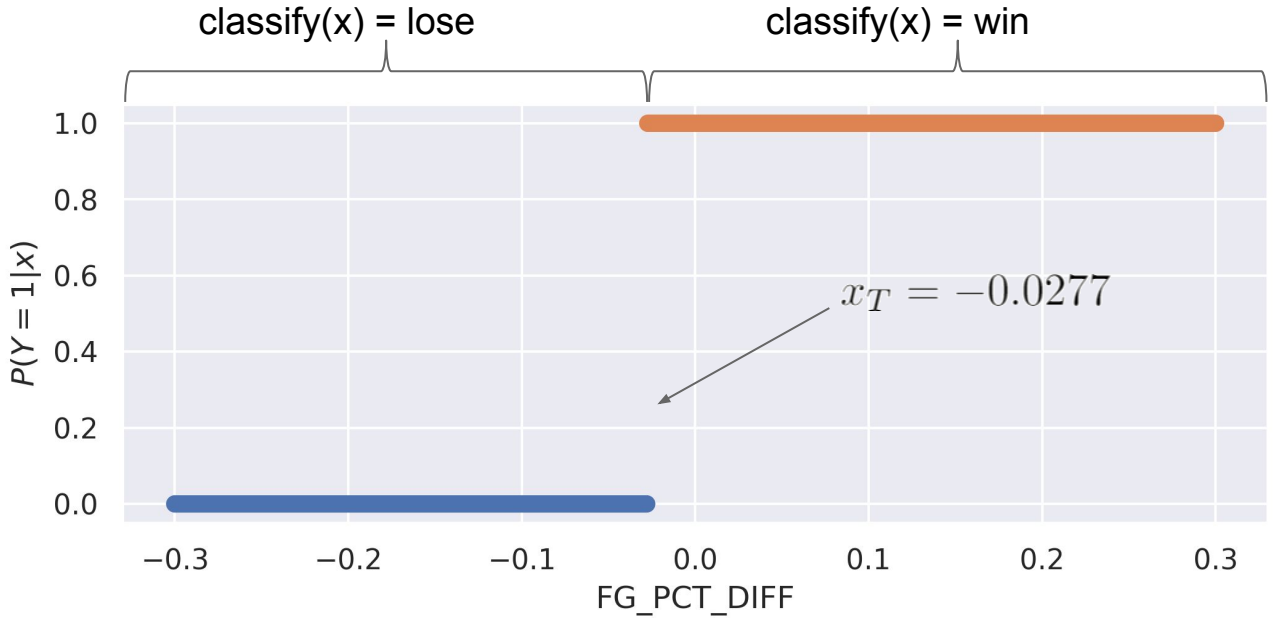
If we pick a threshold, i.e. T = 0.3, we get a predicted class from probabilities.



Our Model

If we pick a threshold, i.e. T = 0.3, we get a predicted class from probabilities.

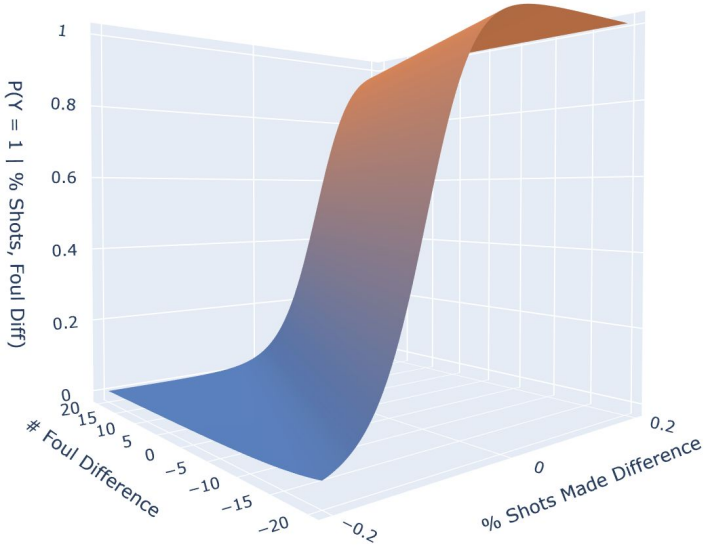
- Effect is that x < x_T predicts class 0, and x > x_T predicts class 1.
- x_T is known as a decision boundary.
  - x_T is a function of model parameters and T.



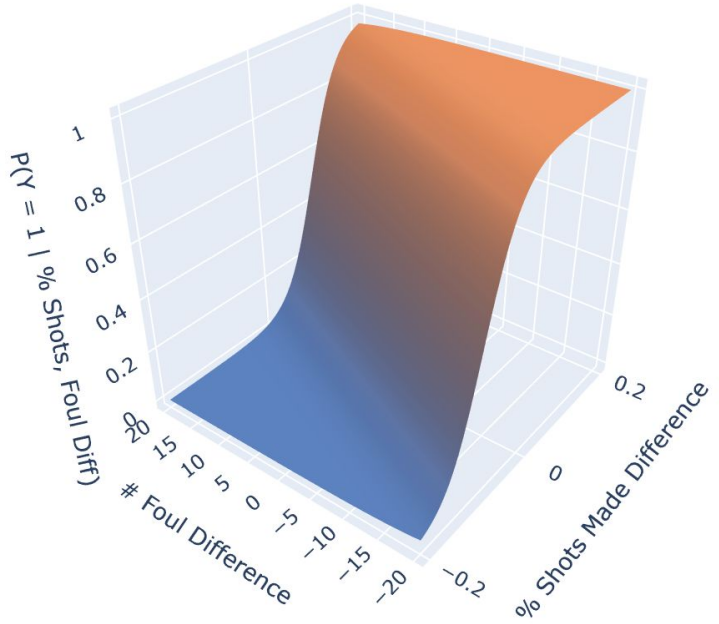
2D Models

Suppose we try to predict the probability of winning from FG_PCT_DIFF and the difference in the number of personal fouls.

- We get a 2D probability surface (same plot from 2 angles shown below).



$$\hat{y} = \sigma(\beta_0 + \beta_1 \text{FG\_PCT\_DIFF} + \beta_2 \text{PF\_DIFF})$$

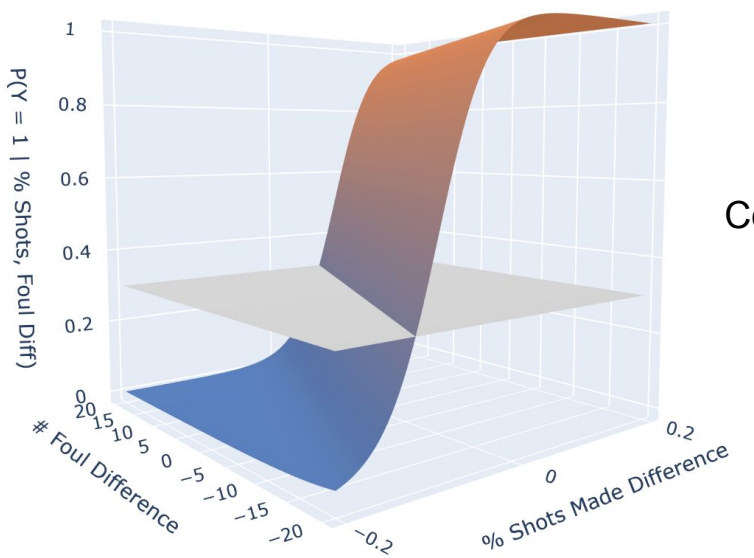


$$\hat{y} = \sigma(\beta_0 + \beta_1 \text{FG\_PCT\_DIFF} + \beta_2 \text{PF\_DIFF})$$

2D Models

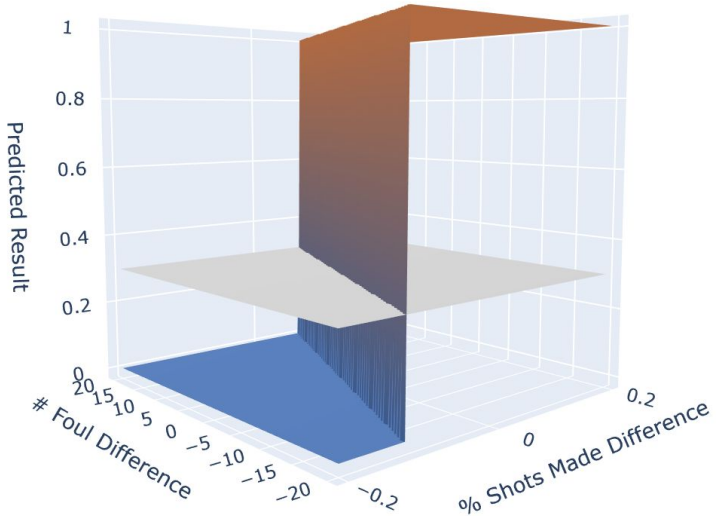
Suppose we try to predict the probability of winning from FG_PCT_DIFF and the difference in the number of personal fouls.

- We get a 2D probability surface.
- To get classes, we again pick a threshold, e.g. T = 0.3.



$$\hat{y} = \sigma(\beta_0 + \beta_1 \text{FG\_PCT\_DIFF} + \beta_2 \text{PF\_DIFF})$$

Compute classes

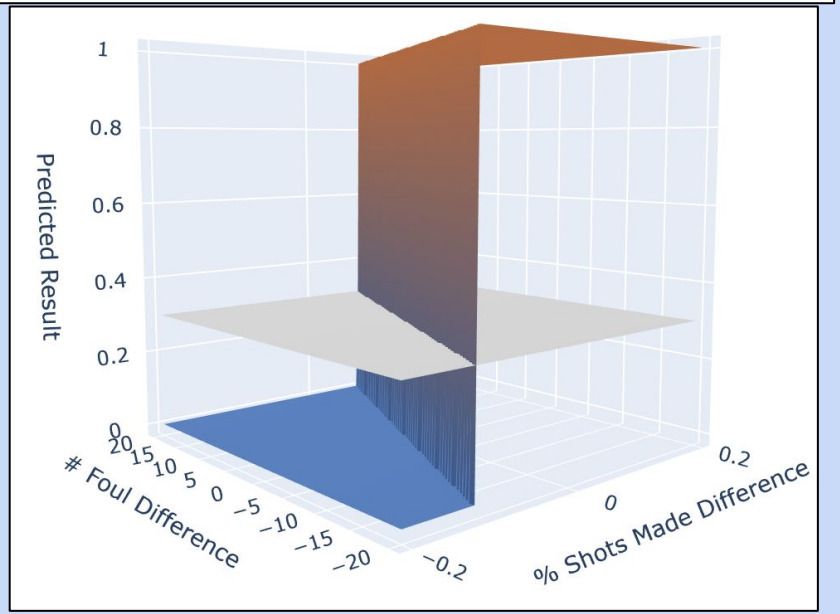


$$\hat{y} = \begin{cases} 1 & \sigma(\beta_0 + \beta_1 \text{FG\_PCT\_DIFF} + \beta_2 \text{PF\_DIFF}) \geq T \\ 0 & \sigma(\beta_0 + \beta_1 \text{FG\_PCT\_DIFF} + \beta_2 \text{PF\_DIFF}) < T \end{cases}$$

2D Models

Consider this thresholded model. Describe the decision boundary.

$$\hat{y} = \begin{cases} 1 & \sigma(\beta_0 + \beta_1 \text{FG\_PCT\_DIFF} + \beta_2 \text{PF\_DIFF}) \geq T \\ 0 & \sigma(\beta_0 + \beta_1 \text{FG\_PCT\_DIFF} + \beta_2 \text{PF\_DIFF}) < T \end{cases}$$

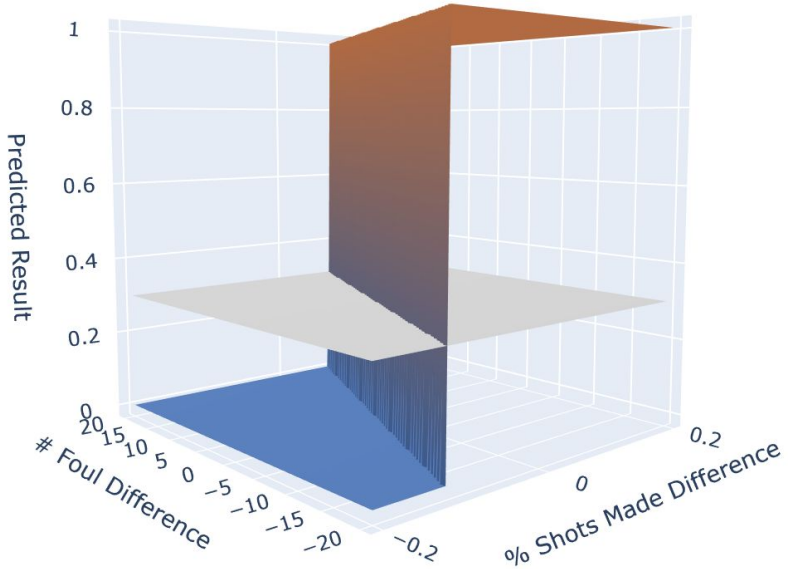


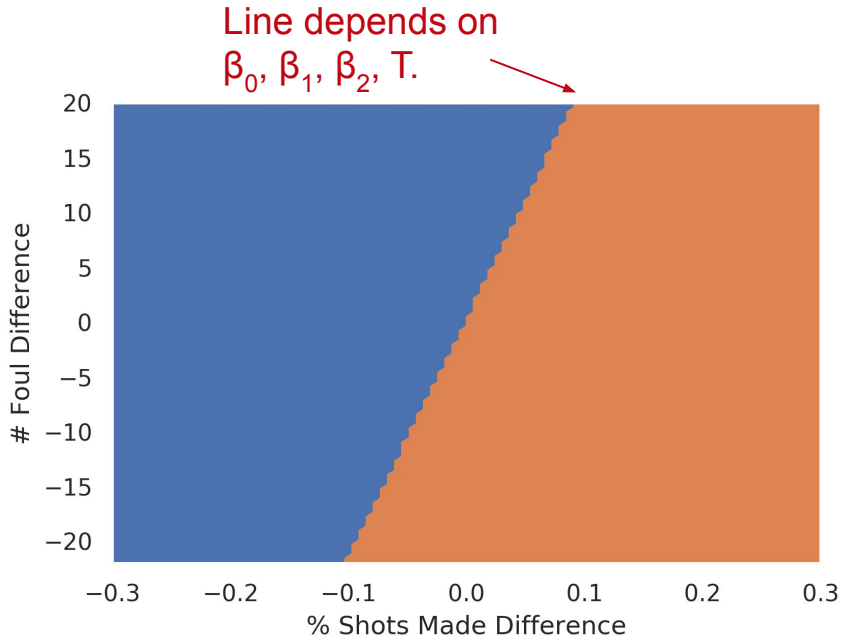
2D Models

Consider this thresholded model. Describe the decision boundary: It's a **line**.

- Everything above the line is predicted to be in class 1.
- Everything below the line is predicted to be in class 0.

$$\hat{y} = \begin{cases} 1 & \sigma(\beta_0 + \beta_1 \text{FG\_PCT\_DIFF} + \beta_2 \text{PF\_DIFF}) \geq T \\ 0 & \sigma(\beta_0 + \beta_1 \text{FG\_PCT\_DIFF} + \beta_2 \text{PF\_DIFF}) < T \end{cases}$$



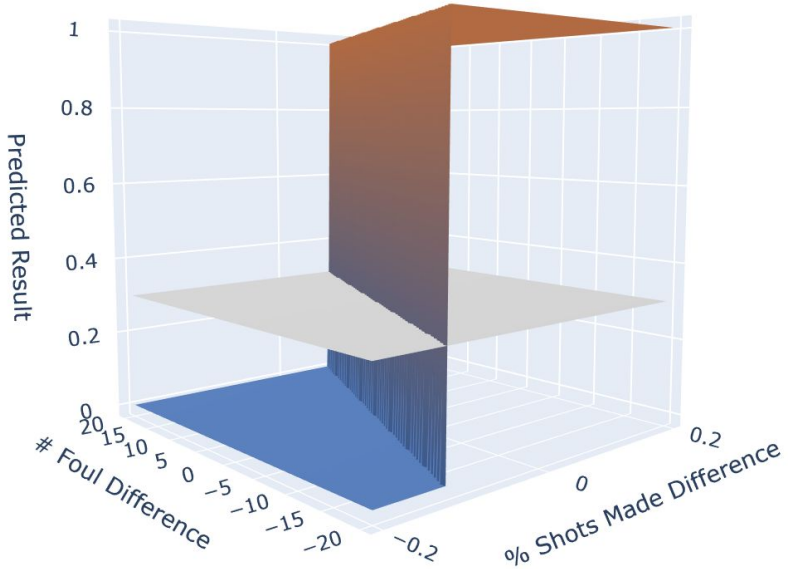


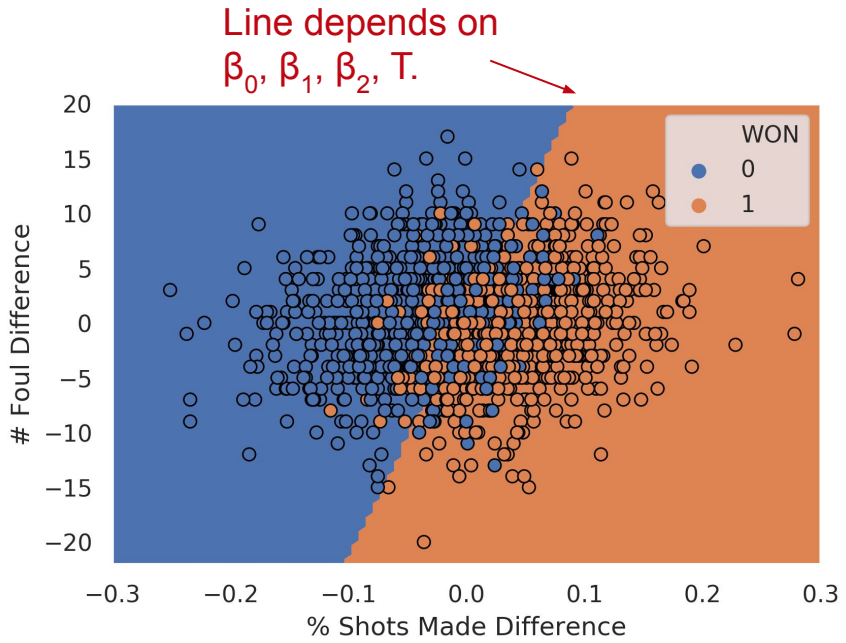
2D Models

Overlaying our model with the data, we can get a rough idea of the accuracy.

- Blue dots on the orange side are false positives (predicted 1, was 0).
- Orange dots on the blue side are false negatives (predicted 0, was 1).

$$\hat{y} = \begin{cases} 1 & \sigma(\beta_0 + \beta_1 \text{FG\_PCT\_DIFF} + \beta_2 \text{PF\_DIFF}) \geq T \\ 0 & \sigma(\beta_0 + \beta_1 \text{FG\_PCT\_DIFF} + \beta_2 \text{PF\_DIFF}) < T \end{cases}$$





Evaluating Classifiers

Classifier Quality

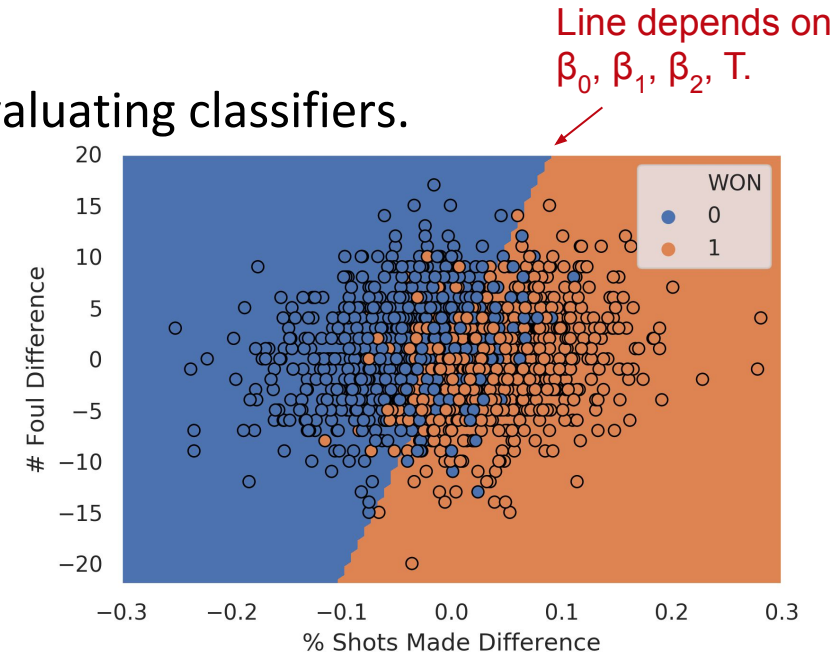
As we adjust our parameters (in this case, β_0, β_1, β_2, T), we get different decision boundaries.

In lecture 20, we focused on 3 metrics for evaluating classifiers.

- Accuracy.
- Precision.
- Recall.

Let's discuss five (!!) more:

- Confusion Matrices (saw in HW7).
- Precision/Recall Curves (saw briefly).
- False Positive Rates, True Positive Rates, and ROC Curves.



Evaluating Classifiers (Review)

Prediction \ Truth	0	1
0	True negative (TN)	False negative (FN)
1	False positive (FP)	True positive (TP)

Some authors (and our textbook) give the opposite ordering for rows and columns, e.g. true positives in the top left instead.

Accuracy: (TP + TN) / n

Error rate: (FP + FN) / n

} The most obvious evaluation metric.

Precision: TP / (TP + FP)

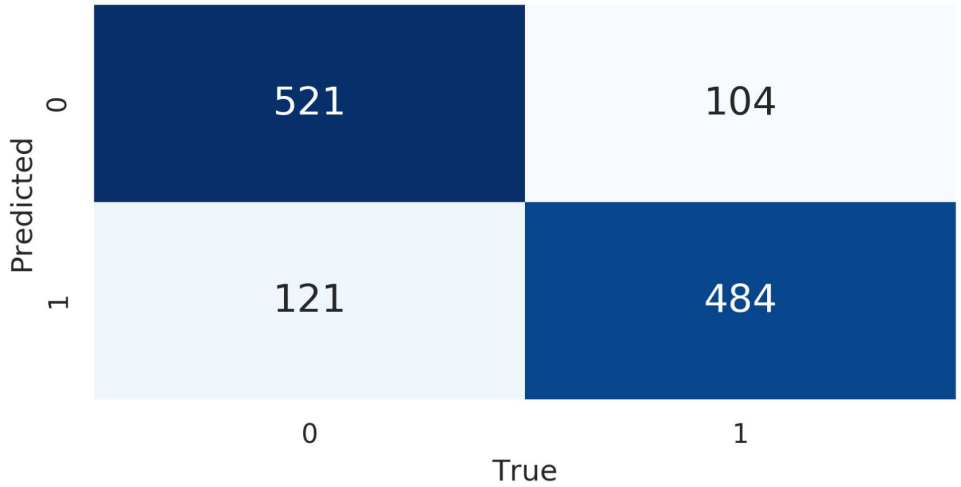
Recall: TP / (TP + FN)

} Used when detecting rare outcomes

Confusion Matrices

Confusion matrix shows predicted vs. true classes:

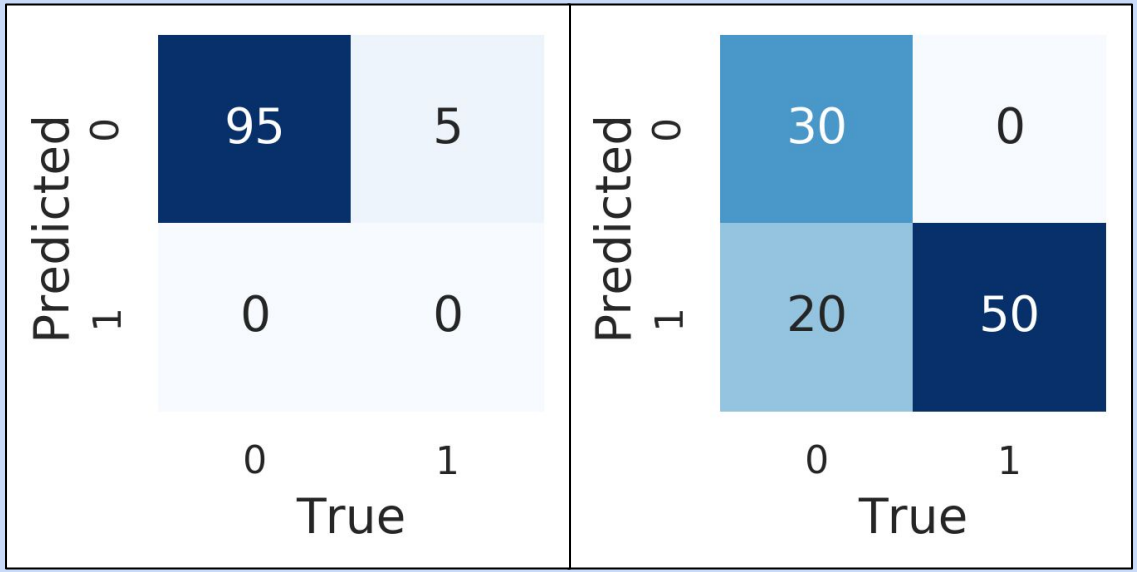
Prediction \ Truth	0	1
0	True negative (TN)	False negative (FN)
1	False positive (FP)	True positive (TP)




```
sns.heatmap(cm, annot=True, fmt = "d", cmap = "Blues", annot_kws={"size": 16})
```

Practice with Metrics

Find accuracy, precision, recall for the following classifiers.



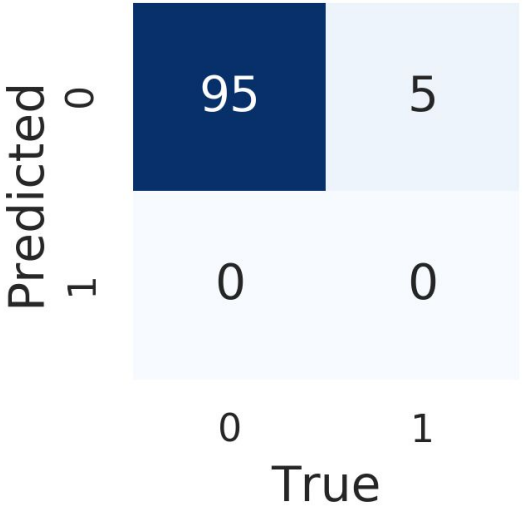
Accuracy: Proportion correct

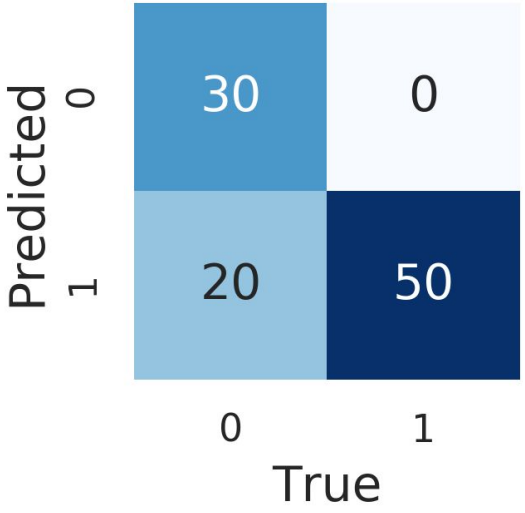
Precision: Of predicted 1s, what prop was actually 1? Aka: how precise were my predictions?

Recall: Of actual 1s, what prop was predicted 1? Aka: how many examples did my classifier recall?

Practice with Metrics

Find accuracy, precision, recall for the following classifiers.





Accuracy:	0.95	0.8
Precision:	Undefined	50/70 = 0.71
Recall:	0	50/50 = 1.00

Accuracy: Proportion correct

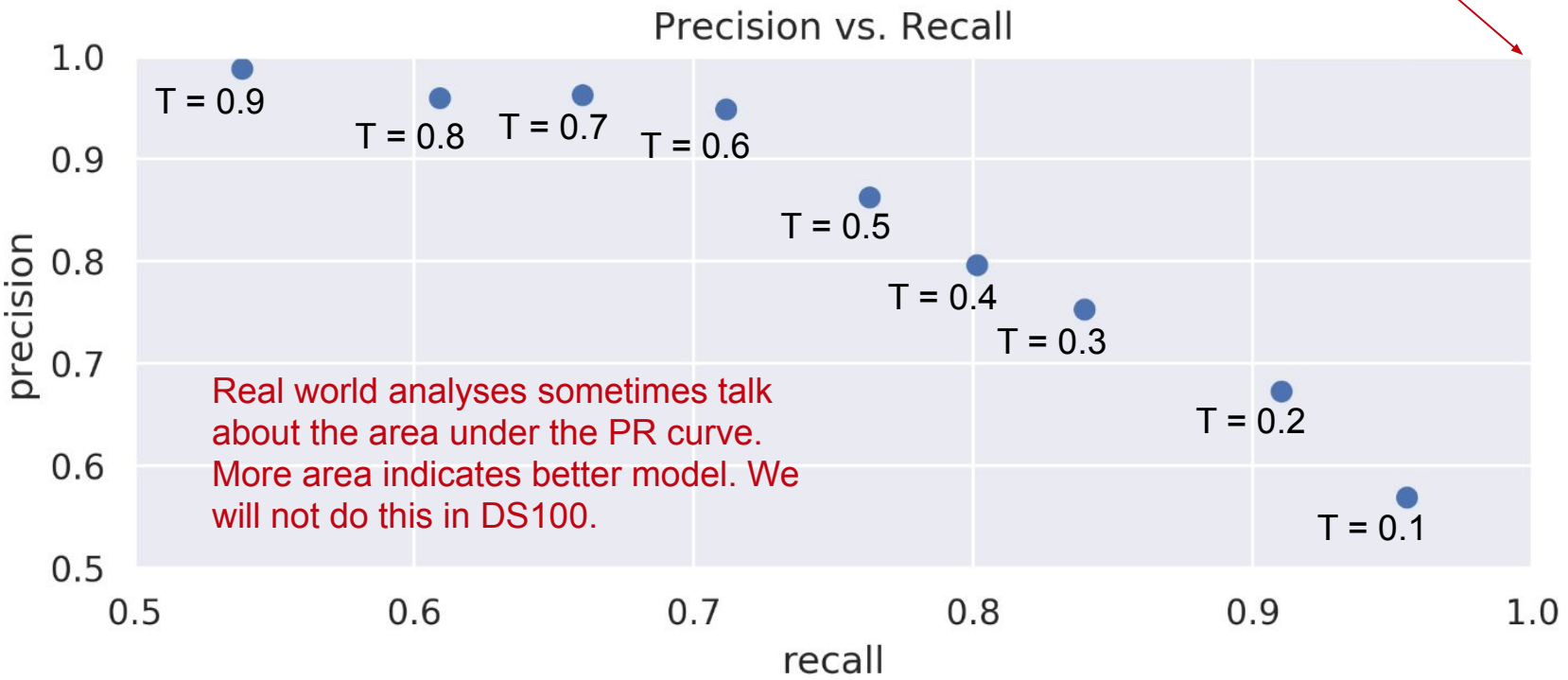
Precision: Of predicted 1s, what prop was actually 1? Aka: how precise were my predictions?

Recall: Of actual 1s, what prop was predicted 1? Aka: how many examples did my classifier recall?

Precision Recall Curves

As we adjust our threshold, our precision and recall will change.

- Very low threshold: High recall but low precision.
- Very high threshold: Low recall but high precision.



Even More Evaluation Metrics: FPR and TPR

		Truth: 0	Truth: 1
Prediction	0	True negative (TN)	False negative (FN)
	1	False positive (FP)	True positive (TP)

False Positive Rate (FPR):

- FP / (FP + TN)
- "What proportion of innocent people did I convict?"

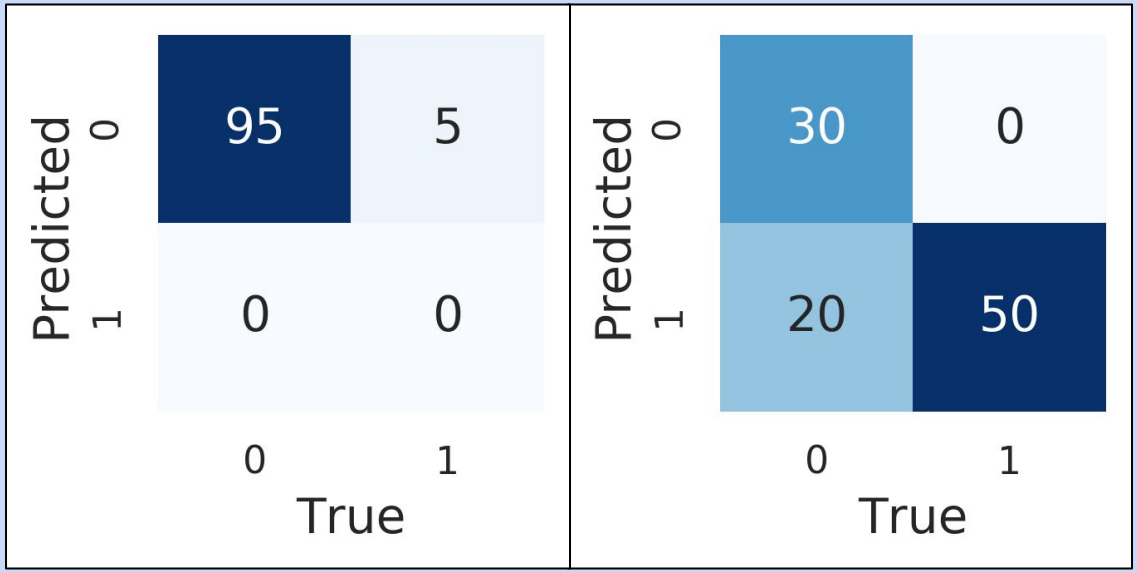
True Positive Rate (TPR):

- TP / (TP + FN)
- "What proportion of guilty people did I convict?"
- Note that TPR = recall.

This may seem like a lot of different assessments for classifiers. In fact, there are many more metrics that we won't cover!

Practice with Metrics

Find TPR and FPR for the following classifiers.

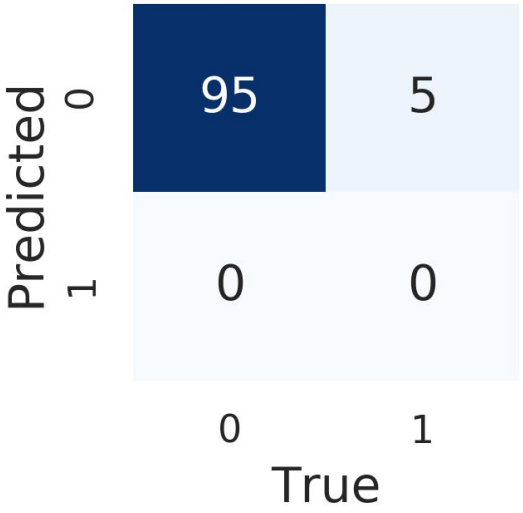


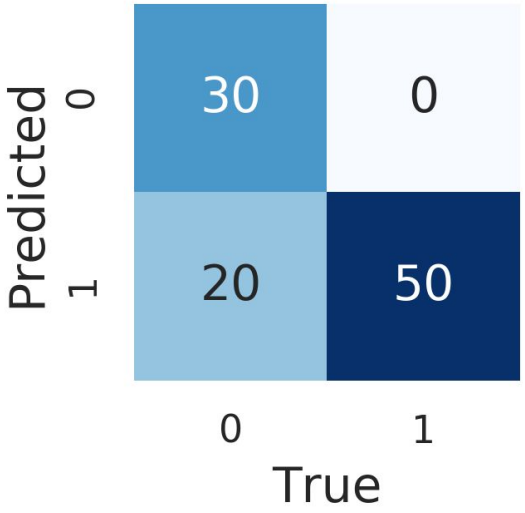
FPR: FP / (FP + TN), or "What proportion of innocent people did I convict?"

TPR: TP / (TP + FN), or "What proportion of guilty people did I convict?"

Practice with Metrics

Find TPR and FPR for the following classifiers.





FPR: 0/95 = 0 20/50 = 0.4

TPR: 0/5 =0 50/50 = 1

FPR: FP / (FP + TN), or "What proportion of innocent people did I convict?"

TPR: TP / (TP + FN), or "What proportion of guilty people did I convict?"

Decision Cutoffs

As with precision/recall, changing threshold also changes FPR and TPR.

If decision rule cutoff is set to 0, then we say everyone is guilty.

If decision rule cutoff is set to 1, then we say everyone is innocent.

What's FPR and TPR for each of these cases?

		Truth: 0	Truth: 1
Prediction	0	True negative (TN)	False negative (FN)
	1	False positive (FP)	True positive (TP)

FPR: FP / (FP + TN), or "What proportion of innocent people did I convict?"

TPR: TP / (TP + FN), or "What proportion of guilty people did I convict?"

Decision Cutoffs

As with precision/recall, changing threshold also changes FPR and TPR.

If decision rule cutoff is set to 0, then we say everyone is guilty. **FPR:** 1, **TPR:** 1

If decision rule cutoff is set to 1, then we say everyone is innocent. **FPR:** 0, **TPR:** 0

What's FPR and TPR for each of these cases?

		Truth	
		0	**1**
Prediction	**0**	True negative (TN)	False negative (FN)
	1	False positive (FP)	True positive (TP)

FPR: FP / (FP + TN), or "What proportion of innocent people did I convict?"

TPR: TP / (TP + FN), or "What proportion of guilty people did I convict?"

ROC Curves

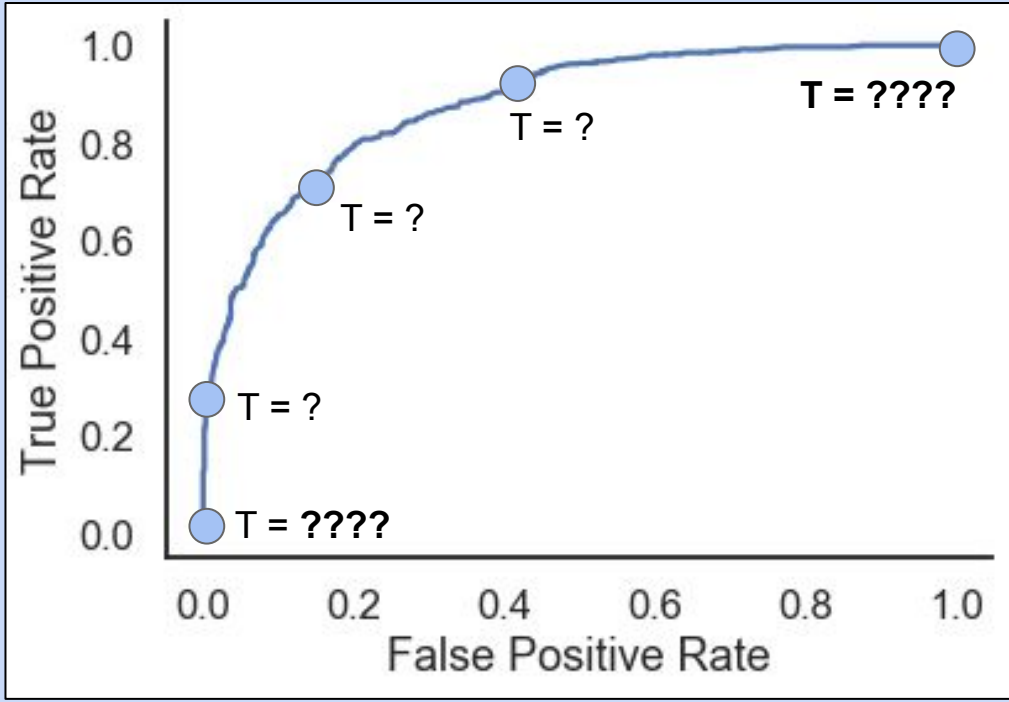
Decreasing the decision threshold:

- Increases TPR (good).
- Increases FPR (bad).

A “ROC curve” shows this tradeoff.

- X-axis: False positive rate.
- Y-axis: True positive rate.

What are the Ts in the bottom left and top right?



ROC Curves

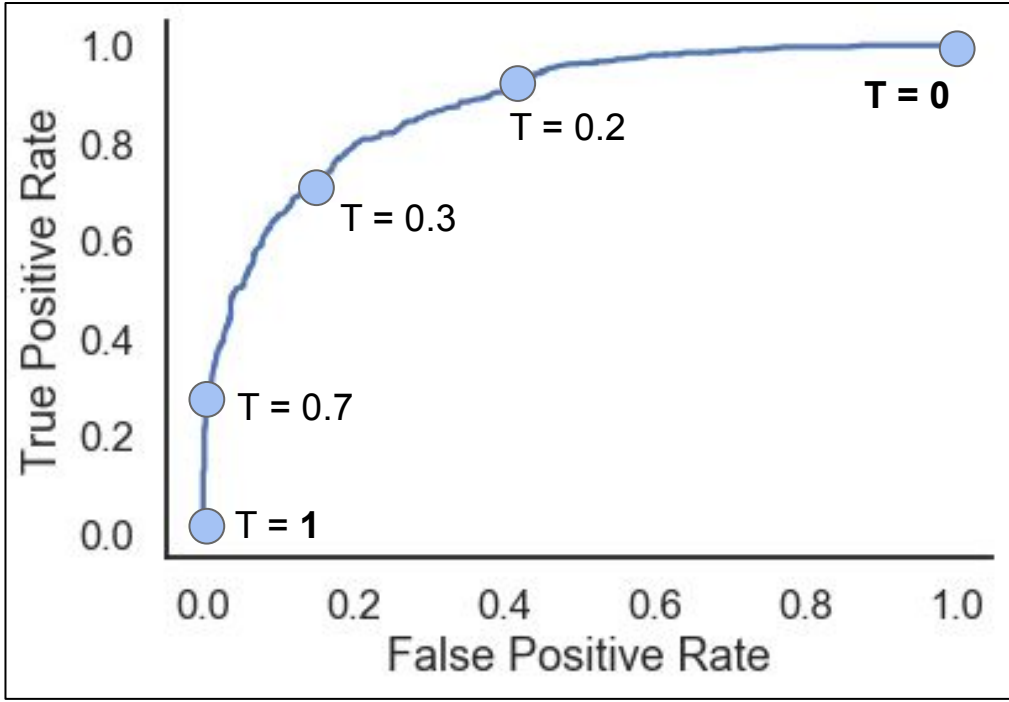
Decreasing the decision threshold:

- Increases TPR (good).
- Increases FPR (bad).

A “ROC curve” shows this tradeoff.

- X-axis: False positive rate.
- Y-axis: True positive rate.

What are the Ts in the bottom left and top right?



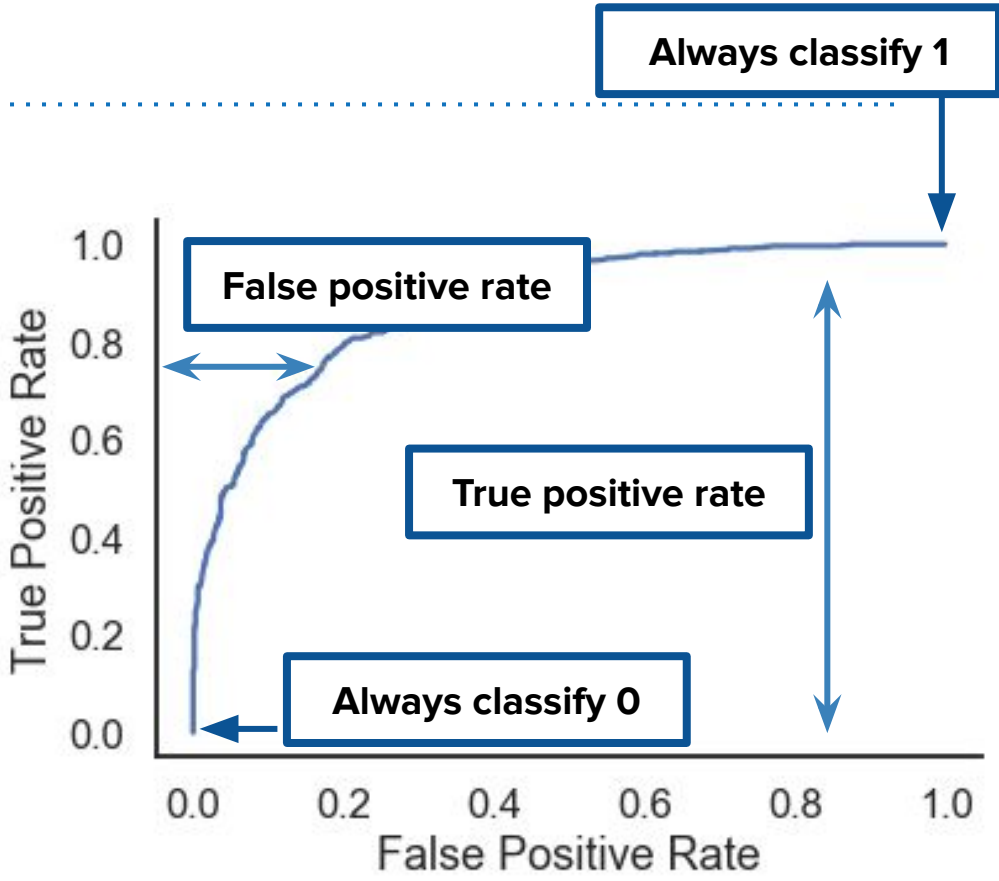
ROC Curves

Decreasing the decision threshold:

- Increases TPR (good).
- Increases FPR (bad).

A “ROC curve” shows this tradeoff.

- X-axis: False positive rate.
- Y-axis: True positive rate.

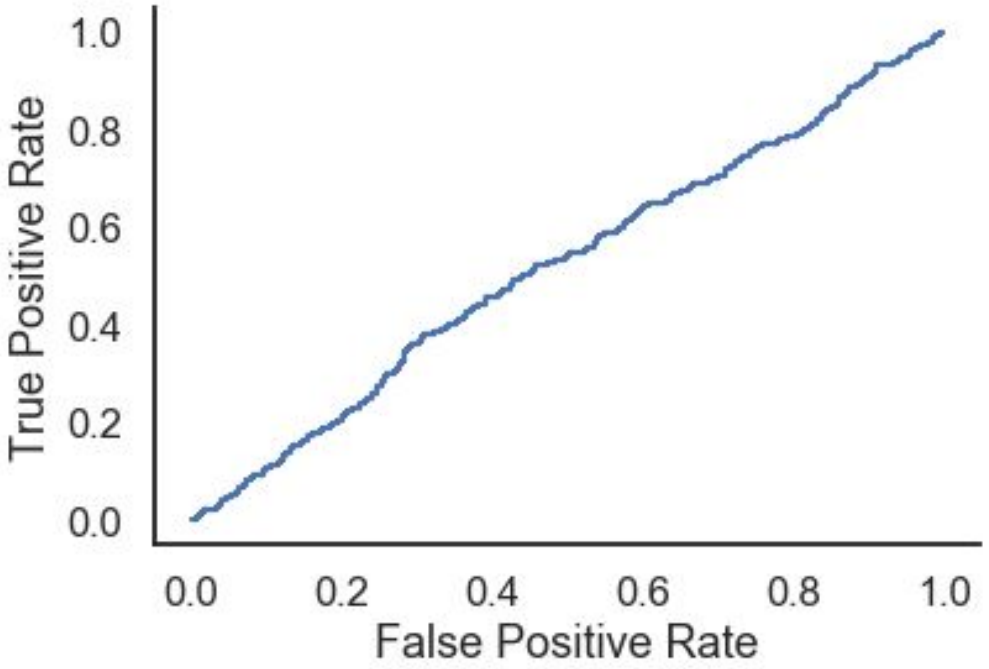


What Good Are ROC Curves?

ROC Curves provide a visual picture of the underlying quality of the regression model.

Example: Random Predictor.

- Assigns random probability between 0 and 1 to any prediction.
- Resulting ROC curve is a diagonal line.

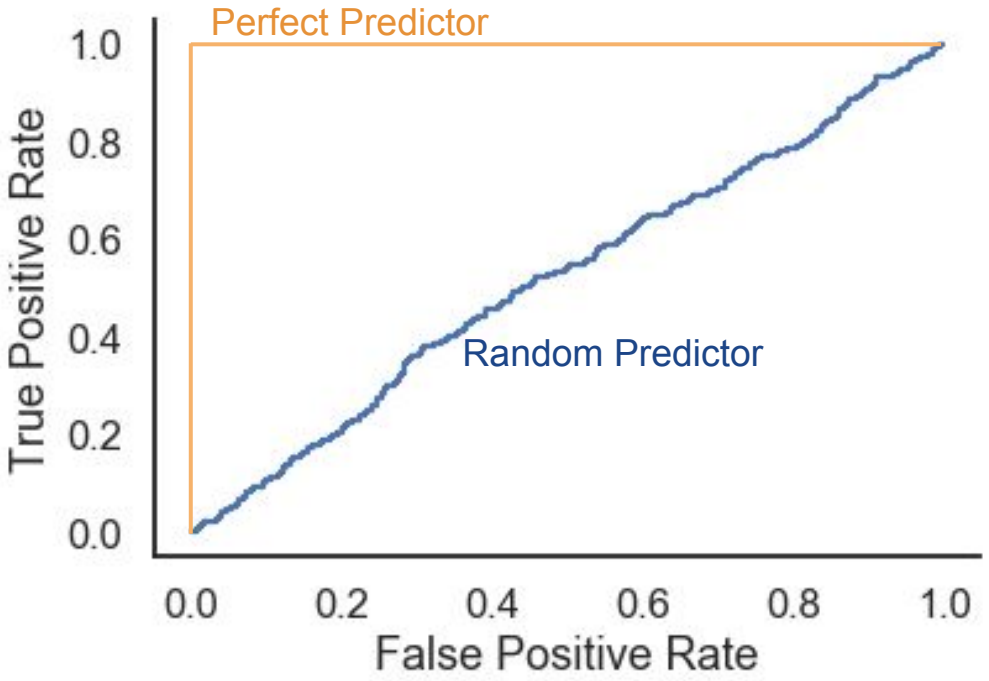


What Good Are ROC Curves?

ROC Curves provide a visual picture of the underlying quality of the regression model.

Example: Perfect Predictor.

- Gives probability 1 to class 1 and probability to class 0.
- Resulting ROC curve is a sharp curve.

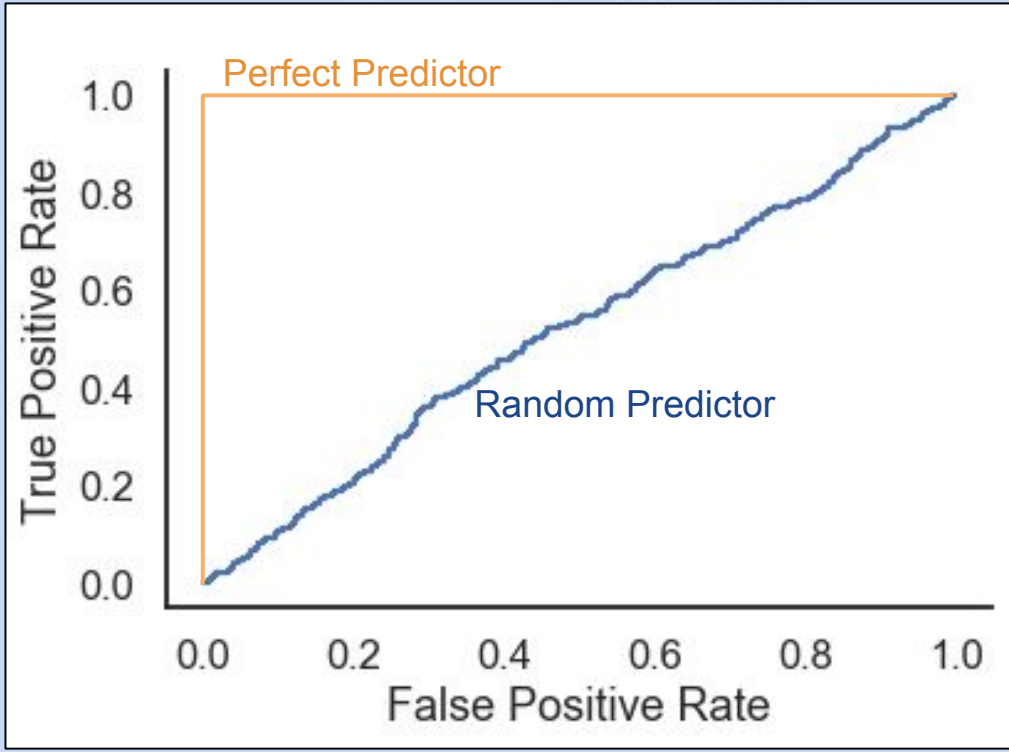


The better a model, the more it will resemble the perfect predictor.

Area under ROC Curve

Can also compute the area under the ROC curve (a.k.a. **AUC**).

- Perfect Predictor: 1
- Random Predictor: ???



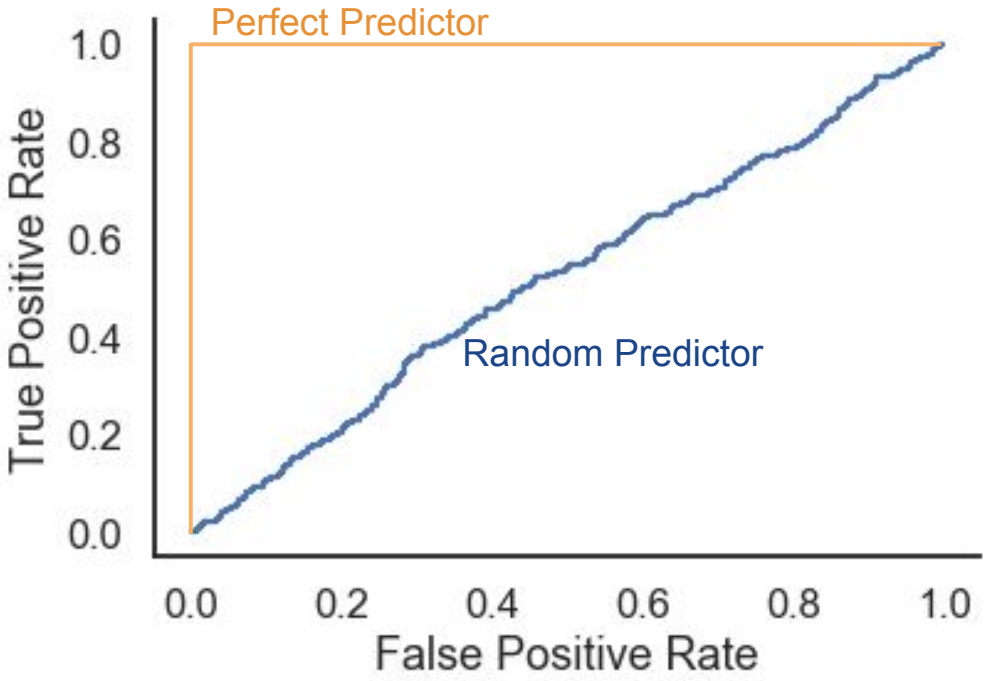
Area under ROC Curve

Can also compute the area under the ROC curve (a.k.a. **AUC**).

- Perfect Predictor: 1
- Random Predictor: 0.5
- Your predictor: Somewhere in [0.5, 1].

The **AUC** provides a dimensionless quantity that characterizes our underlying regression model.

- Dimensionless means it has no units (unlike MSE, mean cross entropy loss, etc.)



It is possible to build a predictor with AUC < 0.5, but that means it does worse than just randomly guessing.

Common Techniques for Evaluating Classifiers

Numerical assessments:

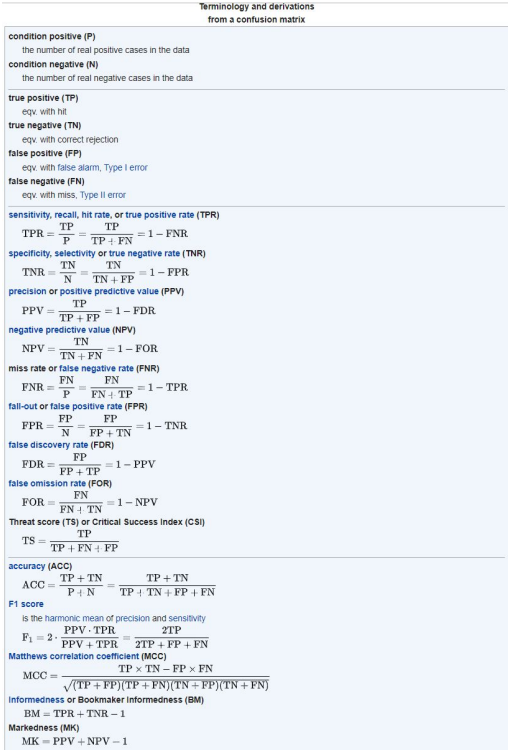
- Accuracy, Precision, Recall, TPR, FPR.
- Area under ROC curve (AUC).

Visualizations:

- Confusion Matrices.
- Precision/Recall Curves.
- ROC Curves.

These 3 are independent of threshold, i.e. evaluate the underlying regression model.

Yes, this seems excessive. [There are actually even more](#)!



Terminology and derivations from a confusion matrix

condition positive (P)
the number of real positive cases in the data
condition negative (N)
the number of real negative cases in the data

true positive (TP)
eqv. with hit
true negative (TN)
eqv. with correct rejection
false positive (FP)
eqv. with false alarm, Type I error
false negative (FN)
eqv. with miss, Type II error

sensitivity, recall, hit rate, or true positive rate (TPR)
$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{P}} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} = 1 - \mathrm{FNR}$
specificity, selectivity or true negative rate (TNR)
$\mathrm{TNR} = \frac{\mathrm{TN}}{\mathrm{N}} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}} = 1 - \mathrm{FPR}$
precision or positive predictive value (PPV)
$\mathrm{PPV} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} = 1 - \mathrm{FDR}$
negative predictive value (NPV)
$\mathrm{NPV} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FN}} = 1 - \mathrm{FOR}$
miss rate or false negative rate (FNR)
$\mathrm{FNR} = \frac{\mathrm{FN}}{\mathrm{P}} = \frac{\mathrm{FN}}{\mathrm{FN} + \mathrm{TP}} = 1 - \mathrm{TPR}$
fall-out or false positive rate (FPR)
$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{N}} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}} = 1 - \mathrm{TNR}$
false discovery rate (FDR)
$\mathrm{FDR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TP}} = 1 - \mathrm{PPV}$
false omission rate (FOR)
$\mathrm{FOR} = \frac{\mathrm{FN}}{\mathrm{FN} + \mathrm{TN}} = 1 - \mathrm{NPV}$
Threat score (TS) or Critical Success Index (CSI)
$\mathrm{TS} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN} + \mathrm{FP}}$

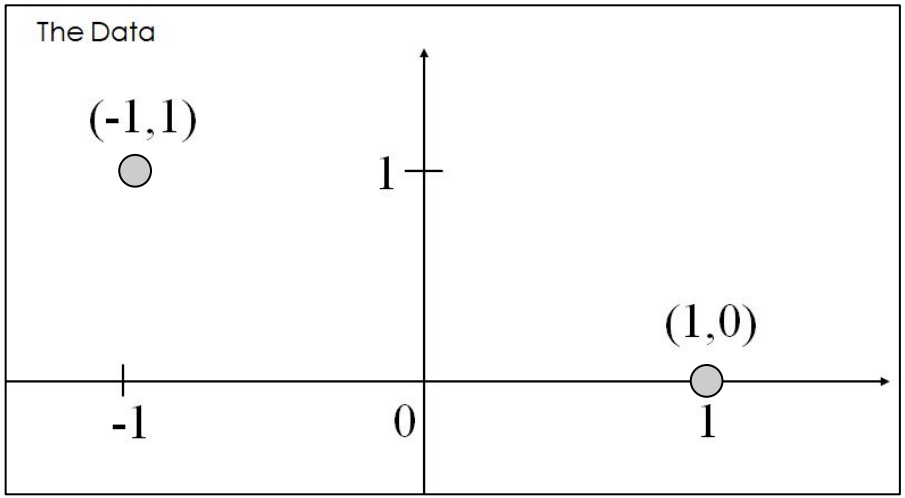
accuracy (ACC)
$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{P} + \mathrm{N}} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}$
F1 score
is the harmonic mean of precision and sensitivity
$\mathrm{F}_1 = 2 \cdot \frac{\mathrm{PPV} \cdot \mathrm{TPR}}{\mathrm{PPV} + \mathrm{TPR}} = \frac{2\mathrm{TP}}{2\mathrm{TP} + \mathrm{FP} + \mathrm{FN}}$
Matthews correlation coefficient (MCC)
$\mathrm{MCC} = \frac{\mathrm{TP} \times \mathrm{TN} - \mathrm{FP} \times \mathrm{FN}}{\sqrt{(\mathrm{TP} + \mathrm{FP})(\mathrm{TP} + \mathrm{FN})(\mathrm{TN} + \mathrm{FP})(\mathrm{TN} + \mathrm{FN})}}$
Informedness or Bookmaker Informedness (BM)
$\mathrm{BM} = \mathrm{TPR} + \mathrm{TNR} - 1$
Markedness (MK)
$\mathrm{MK} = \mathrm{PPV} + \mathrm{NPV} - 1$

Regularization

Question

Suppose we're training a 1 parameter logistic regression model on the data below. What β minimizes the cross-entropy loss?

- $\beta = -\infty$
- $\beta = -1$
- $\beta = 0$
- $\beta = 1$
- $\beta = \infty$



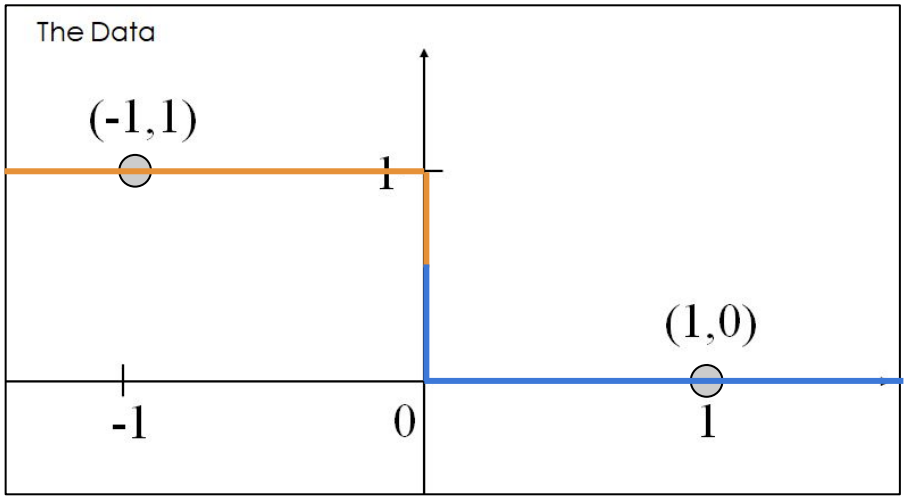
Attendance Question: yellkey.com/

Suppose we're training a 1 parameter logistic regression model on the data below. What β minimizes the cross-entropy loss?

- $\beta = -\infty$

$$\hat{y} = f_{\vec{\beta}}(\vec{x}) = P(Y = 1|\vec{x}) = \sigma(\vec{x}^T \vec{\beta}) \qquad \sigma(t) = \frac{1}{1 + e^{-t}}$$

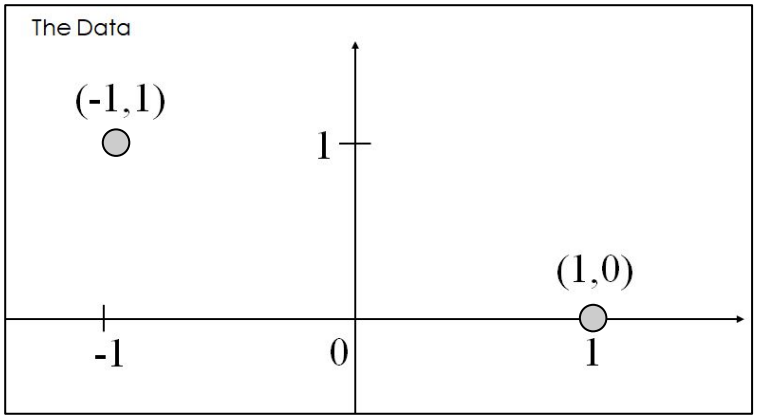
$$\hat{y} = \sigma(-\infty x) = \frac{1}{1 + e^{\infty x}}$$

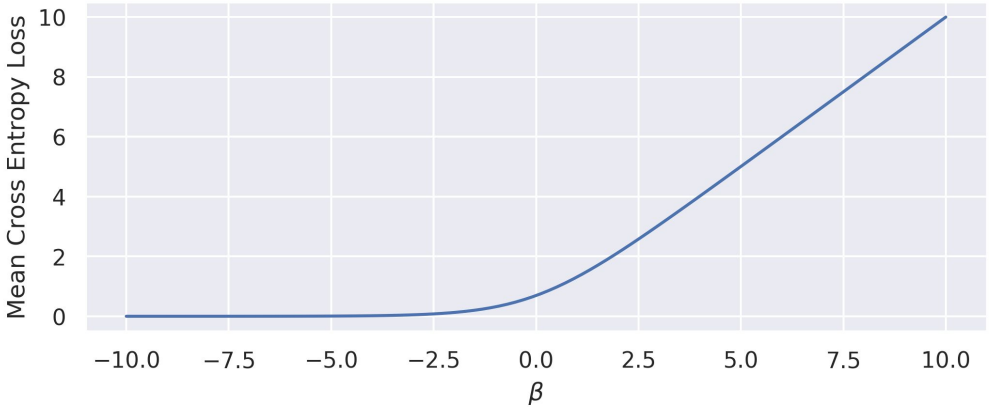


Loss Surface for this Data

On the right is the mean cross entropy loss vs. β for the data on the left.

- Gradient descent will (correctly) push us towards negative infinity.



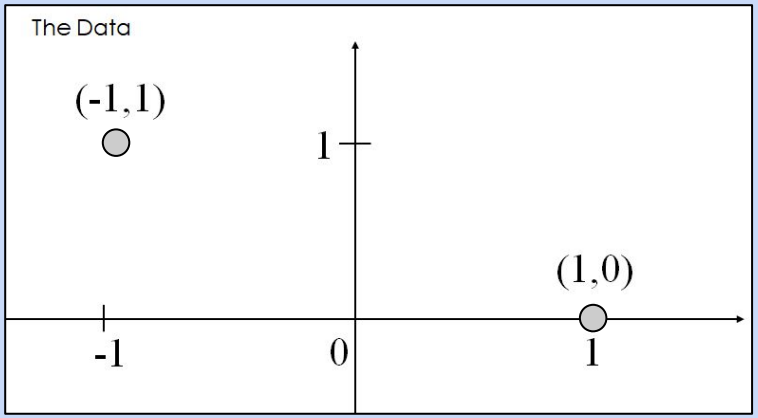


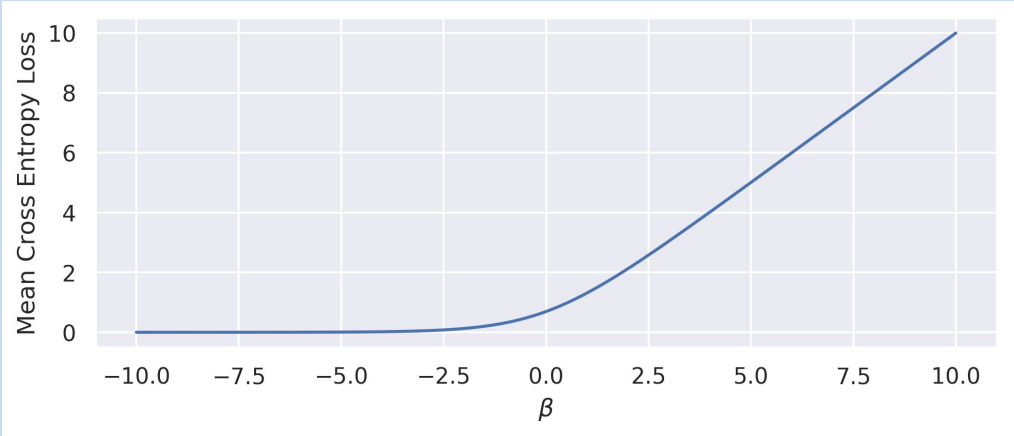
Loss Surface for this Data

On the right is the mean cross entropy loss vs. β for the data on the left.

- Gradient descent will (correctly) push us towards negative infinity.

Give a reason why $\beta = -\infty$ is a good model. Also give a reason why it is a bad model.





Loss Surface for this Data

On the right is the mean cross entropy loss vs. β for the data on the left.

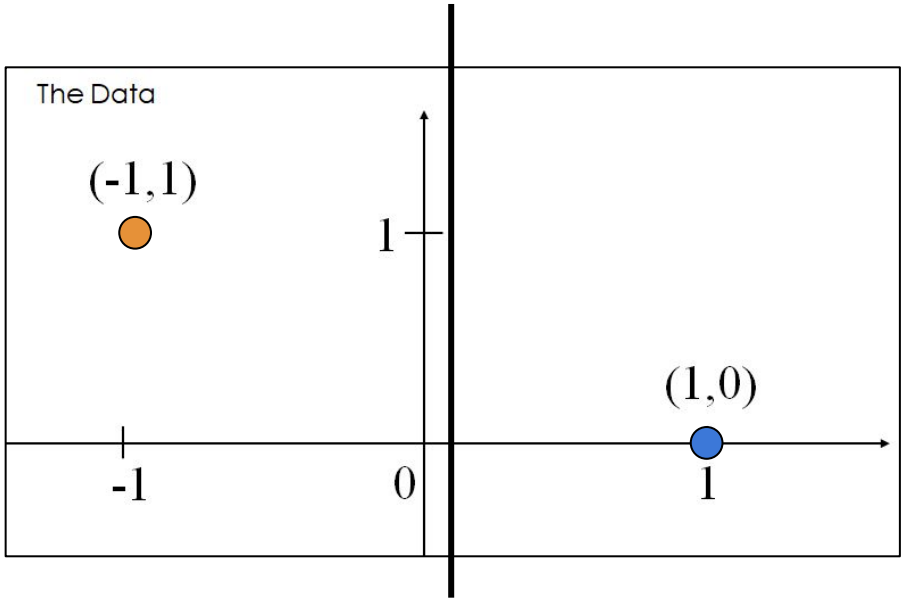
- Gradient descent will (correctly) push us towards negative infinity.

Give a reason why $\beta = -\infty$ is a good model. Also give a reason why it is a bad model.

- Good: It minimizes our loss.
- Bad: It is overconfident. A single misprediction on a test set will yield infinite loss.

Linear Separability

Data that can be separated with a line is called “linearly separable”.

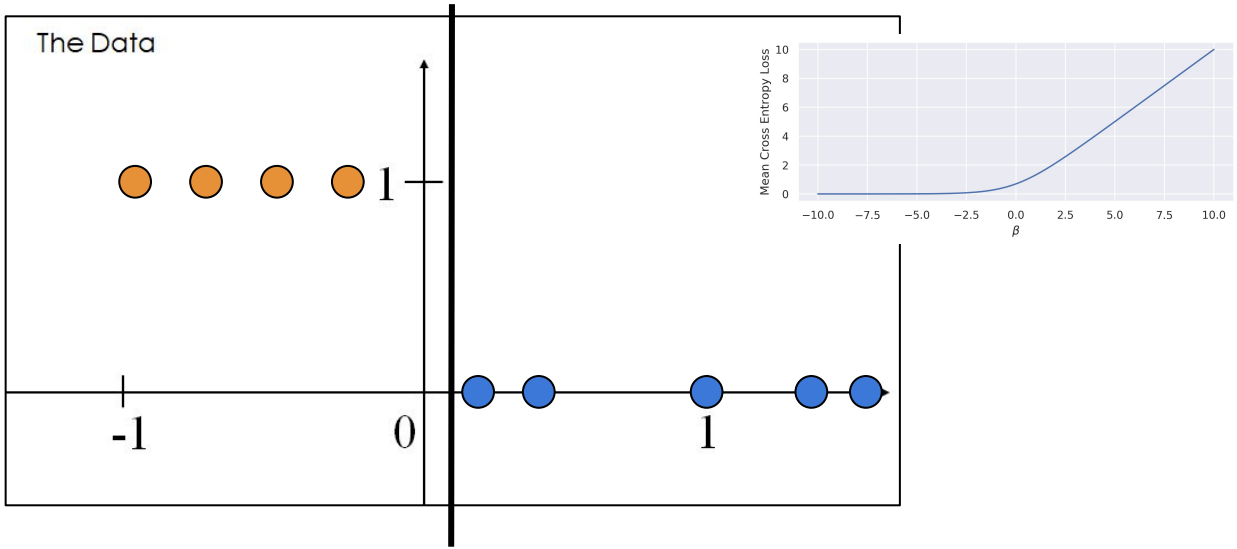


Linear Separability

Data that can be correctly separated with a line is called “linearly separable”.

Logistic regression on linearly separable data will always yield an infinite beta.

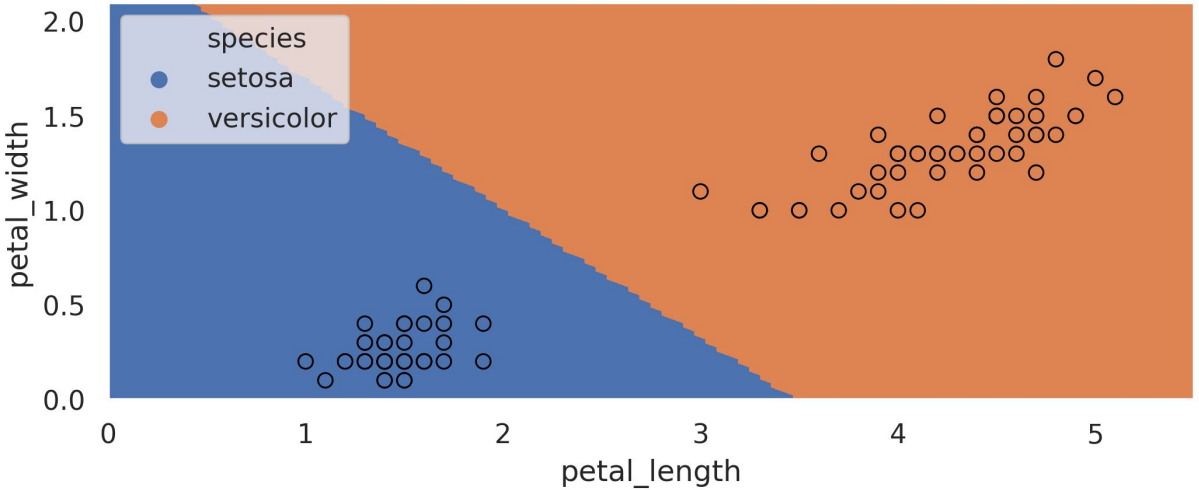
- Gradient descent will roll down the loss surface towards infinity. On a real computer, will stop at an excessively large magnitude β.

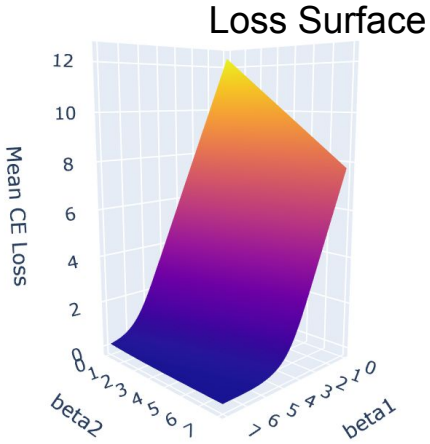


Linear Separability (2D Data)

Below we see a 2D dataset that is linearly separable.

- As before, loss asymptotically approaches 0 for infinite betas.
- When run on a real machine, the example below settled at β_0= -40.5, β_1= 10.77, β_2 = 17.61. These are large betas.



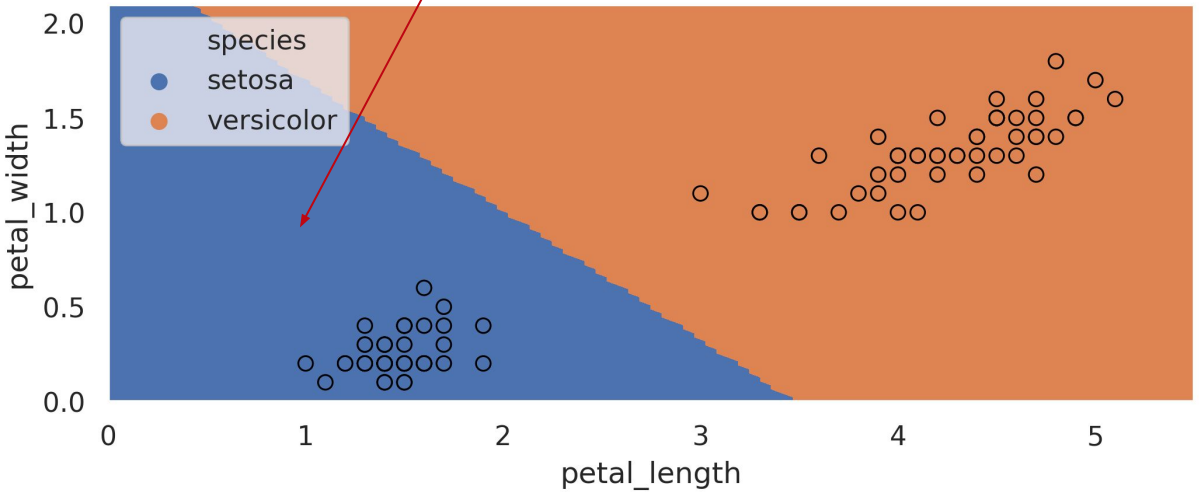


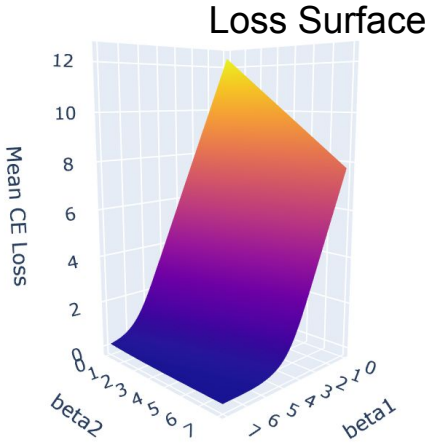
Note: The actual loss surface on the right has 3 independent variables (also has intercept β_0).

Linear Separability (2D Data)

Below we see a 2D dataset that is linearly separable.

- As before, loss asymptotically approaches 0 for infinite betas.
- When run on a real machine, the example below settled at β_0= -40.5, β_1= 10.77, β_2 = 17.61. These are large betas.
  - P(Y = versicolor | length = 1, width = 1) = sigma(-12.2) = 5 x 10^{-6}





Note: The actual loss surface on the right has 3 independent variables (also has intercept β_0).

Avoiding Overfitting: Regularization

Large betas are a form of overfitting.

To avoid overfitting, we know we can use regularization.

- There's nothing new here!

Standard logistic regression:

$$\hat{\beta} = \underset{\vec{\beta}}{\operatorname{argmin}} \frac{1}{n} \sum_i \ell(y_i, \sigma(\vec{x}_i^T \vec{\beta}))$$

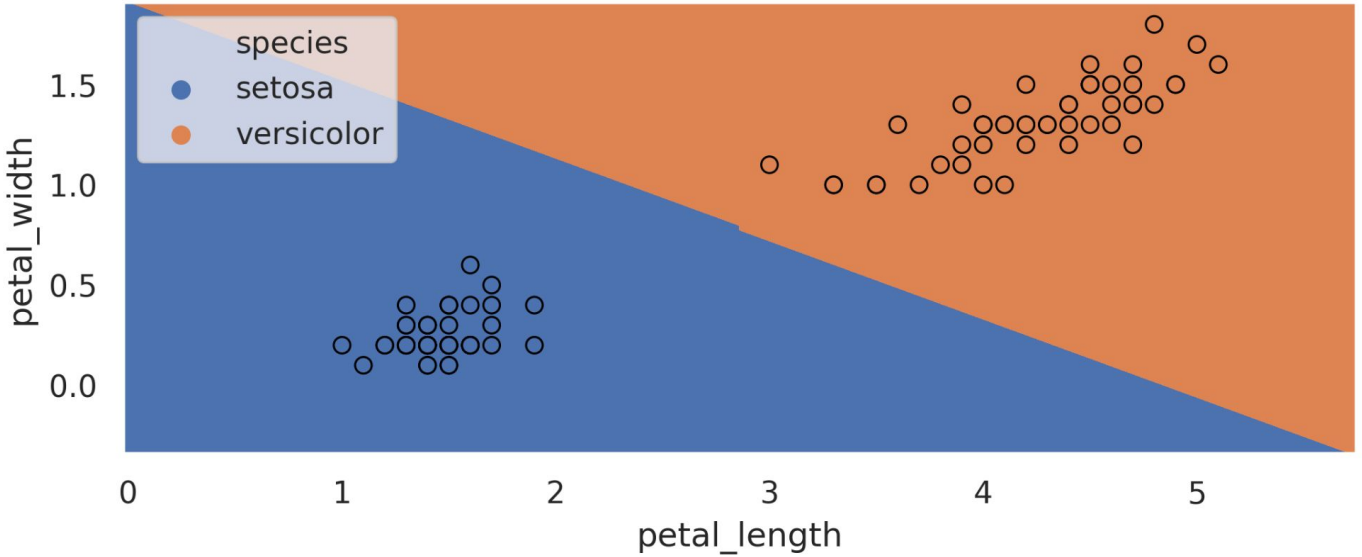
Regularized logistic regression:

$$\hat{\beta} = \underset{\vec{\beta}}{\operatorname{argmin}} \frac{1}{n} \sum_i \ell(y_i, \sigma(\vec{x}_i^T \vec{\beta})) + \lambda \mathcal{S}(\vec{\beta})$$

Results of a Regularized Model

Below: We see results of a regularized model (lambda = 0.2).

- $\beta_0 = 0$, $\beta_1 = 1.07$, $\beta_2 = 1.05$.
- As with linear regression, it's important to put data into standard units before applying regularization.
- Will see the precise details in our demo (coming up next).



Logistic Regression from Scratch (Demo)

Multiclass Classification (Extra)

Multiclass Classification

This topic was not covered in class.

If you'd like to learn more, see:

- CS188
- CS189
- CS182
- Stat 151A
- Info 254

Multiclass Classification

Sometimes we have more than one class that we're interested in.

Example, we want to predict who a democratic voter will vote for:

- Bernie Sanders
- Elizabeth Warren
- Joe Biden
- Andrew Yang
- Tulsi Gabbard
- Amy Klobuchar
- Kamala Harris
- Cory Booker

Order picked based on rankings in first link I clicked on on fivethirtyeight.com.

Multiclass Classification: One vs. Rest

The simplest way to do multiclass classification is to build N binary classifiers, one for each category.

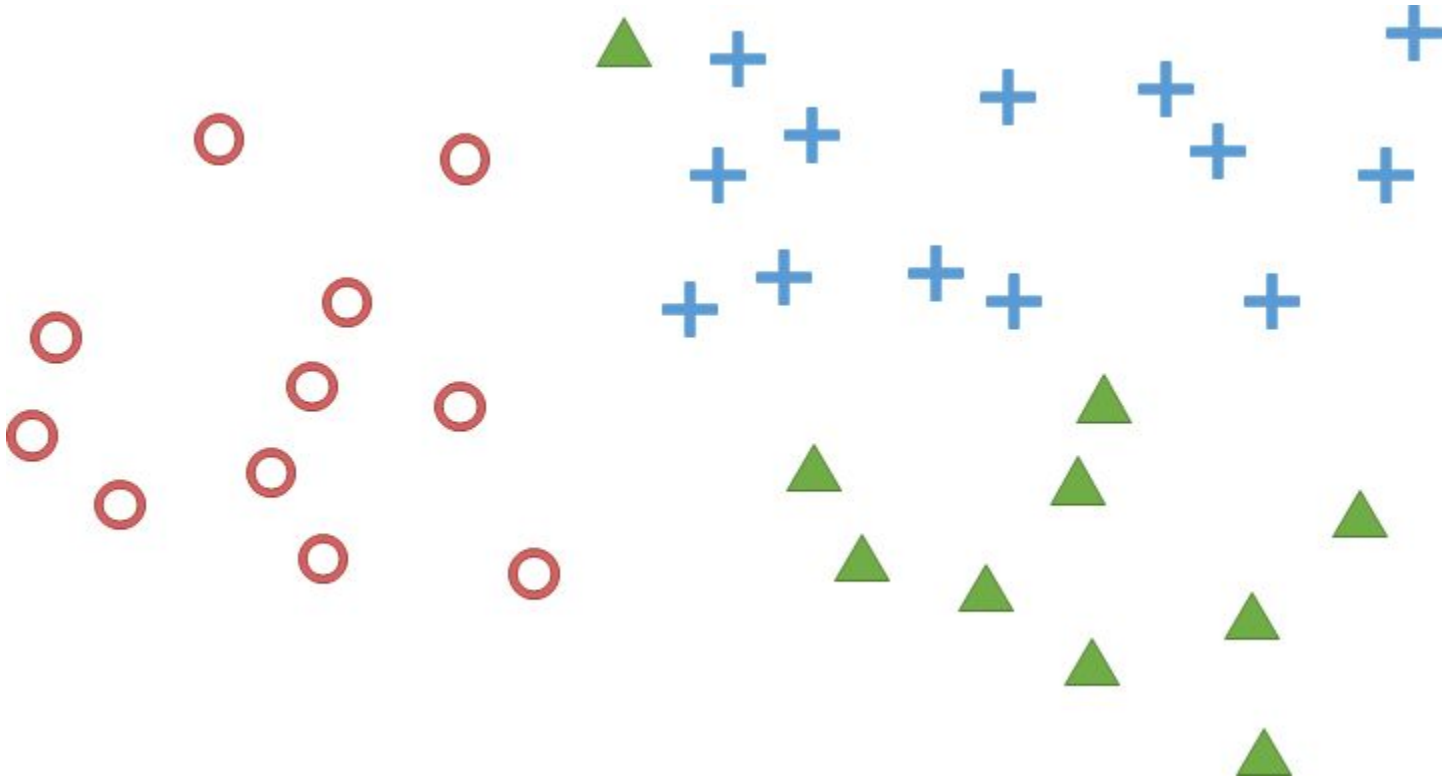
- Resulting prediction will just be whichever classifier gives highest probability.

Example from before:

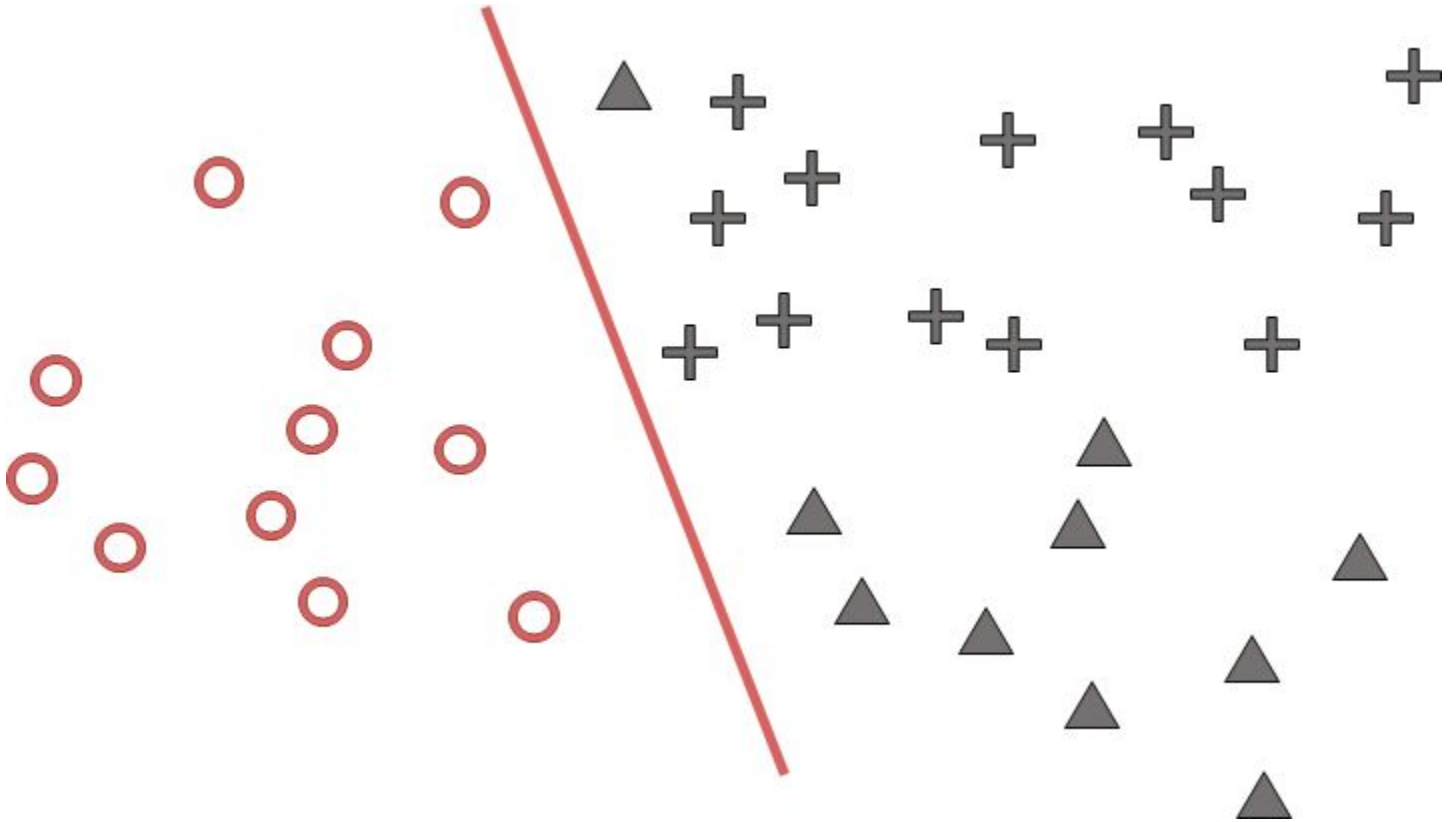
- Build a Bernie Sanders classifier.
- Build an Elizabeth Warren classifier.
- Build a Joe Biden classifier.
- Build an Andrew Yang classifier…

Given a voter, assign the class which has the highest probability among all N.

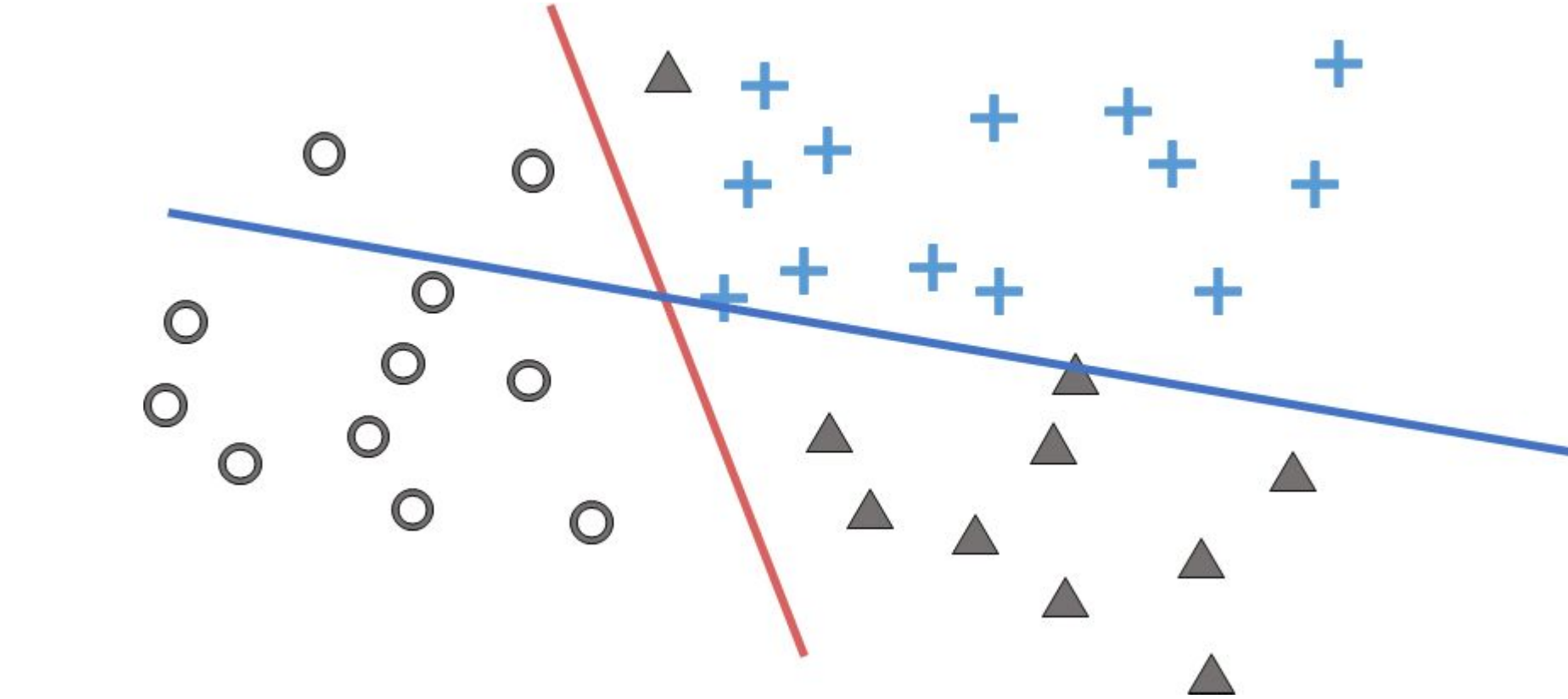
Visual Example



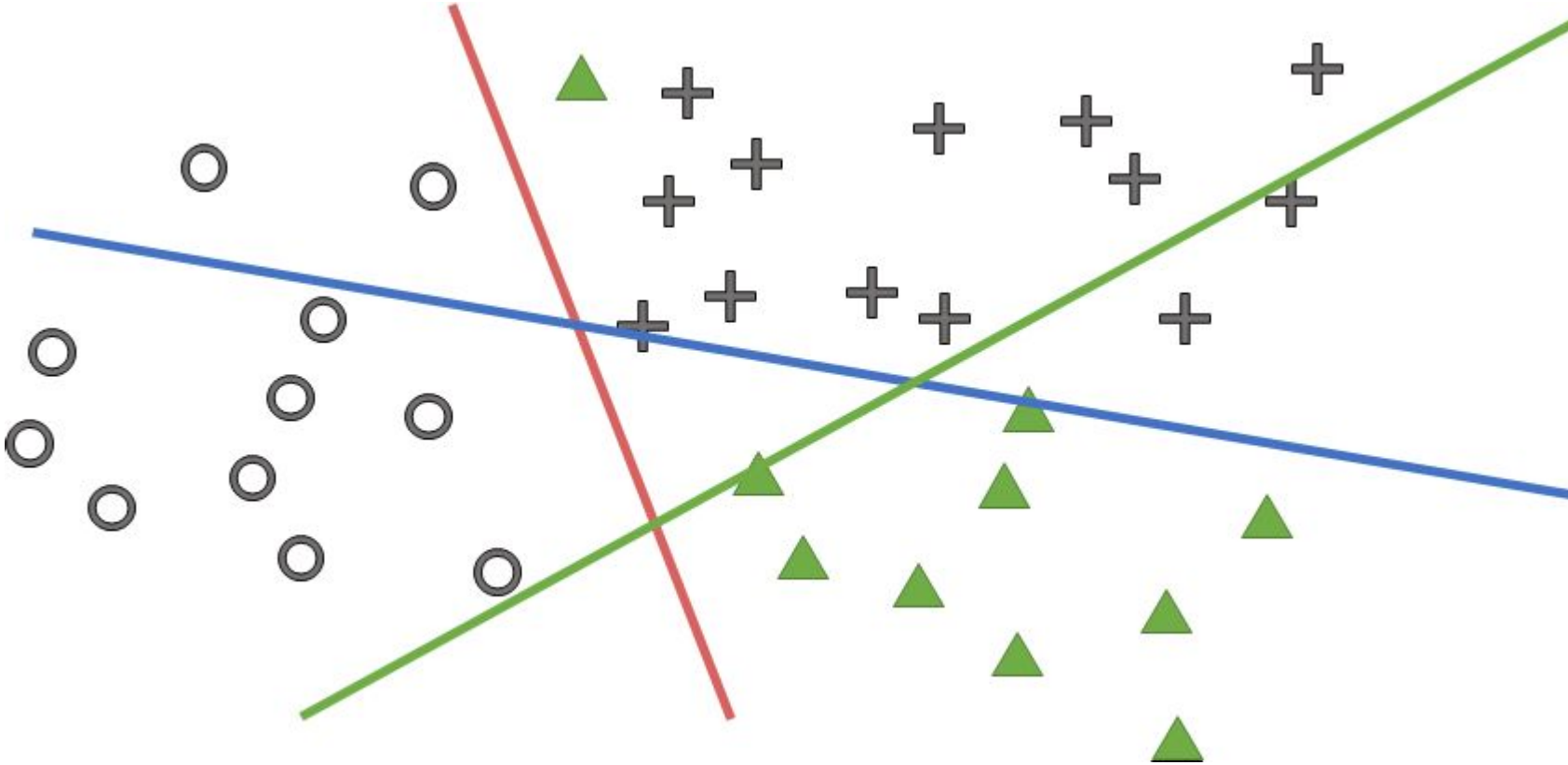
Visual Example



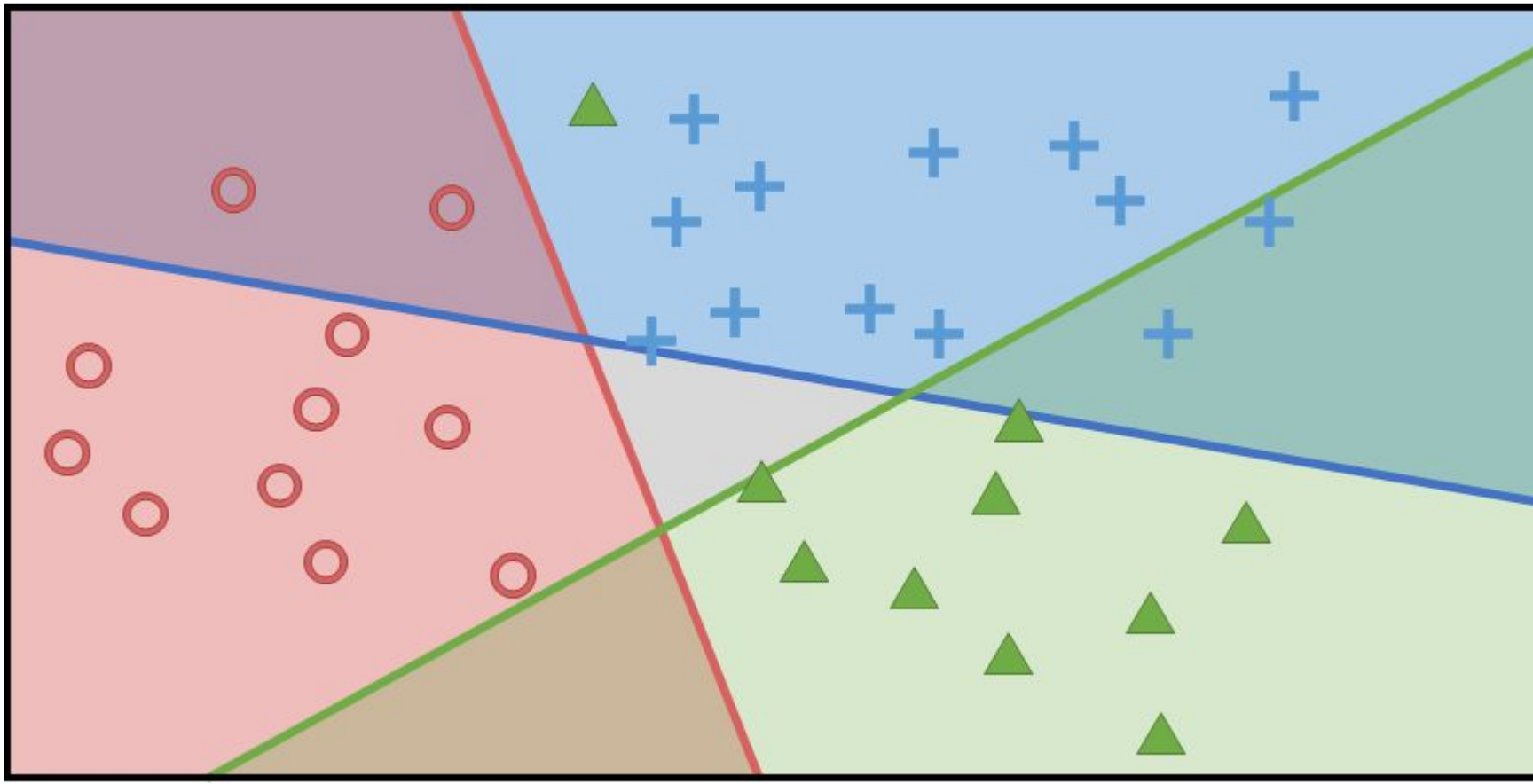
Visual Example



Visual Example



Visual Example



Multiclass Classification: Softmax

One downside of building N binary classifiers: Class imbalance.

Alternate techniques exist that we will not discuss.

Example: Softmax.

- Related to neural networks.
- Idea: Different theta for every class, i.e. for class j we have $\theta^{(j)}$.
- Won't discuss in DS100. See CS188, CS189, CS182, Info 254, or Stat 151A.

$$\mathbf{P}\left(Y = j \,|\, x\right) = \frac{\exp\left(x^T \theta^{(j)}\right)}{\sum_{m=1}^{k} \exp\left(x^T \theta^{(m)}\right)}$$

Summary

Given a logistic regression model, we convert probabilities to classes with a threshold.

- The boundary in the input space between class 0 and class 1 is known as the “decision boundary”.
- In logistic regression, the decision boundary is always linear.
  - In 1D: Decision boundary is a point. In 2D: a plane. In 3D: a space, etc.
  - Here dimensionality is number of inputs, e.g. 2D means two features.
- If data can be perfectly separated by a linear decision boundary, it is “linearly separable”.
  - Will result in very large betas for logistic regression.

Summary

To avoid overfitting a classifier, use regularization.

- Exactly like using regularization with linear regression.
- Don’t forget to scale your data!

Many ways to evaluate classifier quality:

- Numerical:
  - Accuracy, Precision, Recall, TPR, FPR.
  - Area under ROC curve (AUC).
- Visual:
  - Confusion Matrix.
  - Precision/recall Curve.
  - ROC Curve.