

Lecture 3

Semester research project

Goal: Learn and apply a new topic (beyond what we cover in 711/712)

Due Monday (your group should collectively email me):

- Group members and tentative roles
- Tentative topic (ok if this changes for the first few weeks).

Include:

- A couple references
- Why you want to explore this topic

Confusion matrix

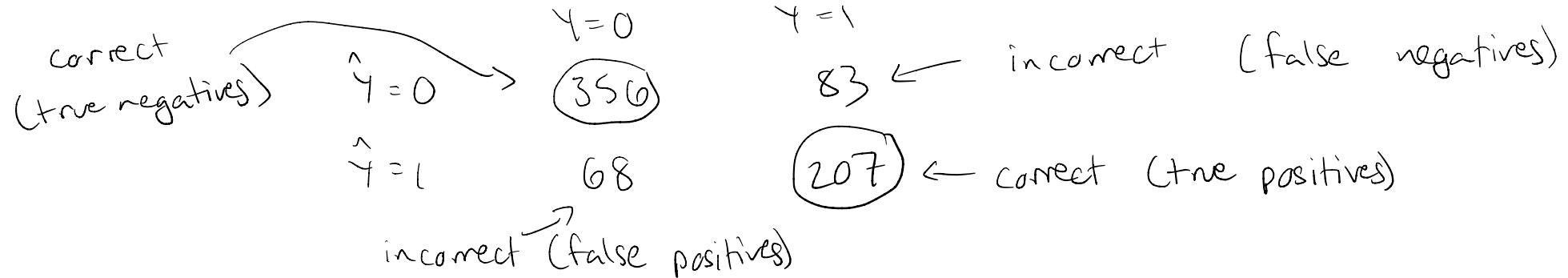
```
1 m1 <- glm(Survived ~ Sex + Age + Pclass, data = titanic,  
2 family = binomial)  
3  
4 table("Predicted" = ifelse(m1$fitted.values > 0.5, 1, 0),  
5 "Observed" = m1$y)
```

Observed			
Predicted	0	1	
0	356	83	
1	68	207	

Confusion matrix

Accuracy: if I randomly select an observation, what is the probability my prediction is correct?

Question: Did we do a good job at predicting survival?



$$\text{Accuracy} = \frac{\# \text{ correct predictions}}{\# \text{ observations}} = \frac{TP + TN}{n} = \frac{356 + 207}{714} = 0.79$$

$$1 - \text{Accuracy} = \text{classification error}$$

Why a threshold of 0.5?

Consider data (x, y)

$$x \in \mathbb{R}^d$$

$$y \in \{0, 1\}$$

$$p(x) = P(Y=1 | X=x) \quad (\text{what we want to estimate})$$

Binary predictions: $\hat{y} = \mathbb{1}\{p(x) \geq h\}$ threshold

(Population) Classification error: $P(Y \neq \hat{y})$ ($= 1 - \text{accuracy}$)

Claim: $h=0.5$ minimizes classification error among all binary classifiers

Proof: Let $c: \mathbb{R}^d \rightarrow \{0, 1\}$ be a binary classifier

$$\hat{y} = c(x)$$

$$\begin{aligned} P(Y \neq \hat{y}) &= P(Y \neq c(x)) = \mathbb{E}[\mathbb{1}\{c(x) \neq Y\}] \\ &= \mathbb{E}[\mathbb{E}[\mathbb{1}\{c(x) \neq Y\} | X]] \end{aligned}$$

(iterated
expectations)

$$\mathbb{E} \left[\mathbb{E} \left[\underbrace{\mathbb{1}\{C(x) \neq Y\}}_{= \begin{cases} \mathbb{E}[\mathbb{1}\{Y=0\}] & C(x)=1 \\ \mathbb{E}[\mathbb{1}\{Y=1\}] & C(x)=0 \end{cases}} \mid X \right] \right]$$

$$= \begin{cases} \mathbb{E}[\mathbb{1}\{Y=0\}] & C(x)=1 \\ \mathbb{E}[\mathbb{1}\{Y=1\}] & C(x)=0 \end{cases} = \mathbb{E}[\mathbb{1}\{Y=0\}] C(x) + \mathbb{E}[\mathbb{1}\{Y=1\}] (1-C(x))$$

$$\rightarrow \mathbb{E} \left[\mathbb{E}[\mathbb{1}\{Y=0\}] C(x) + \mathbb{E}[\mathbb{1}\{Y=1\}] (1-C(x)) \right]$$

$$= \mathbb{E} \left[p(Y=0) C(x) + p(Y=1) (1-C(x)) \right]$$

$$= \mathbb{E}[(1-p(x)) C(x) + p(x) (1-C(x))]$$

$$p(Y \neq \hat{Y}) = \int_x \left[(1-p(x)) C(x) + p(x) (1-C(x)) \right] f(x) dx$$

To minimize $p(Y \neq \hat{Y})$, make integrand as small as possible for each x

$$P(Y \neq \hat{Y}) = \int_x \left[(1-p(x))c(x) + p(x)(1-c(x)) \right] f(x) dx$$

To minimize $P(Y \neq \hat{Y})$, make integrand as small as possible for each x

\Rightarrow want $(1-p(x))c(x) + p(x)(1-c(x))$ to be small

If $1-p(x) \leq p(x)$: $c(x) = 1$

$1-p(x) > p(x)$: $c(x) = 0$

\Rightarrow if $p(x) \geq 0.5$: $c(x) = 1$

$p(x) < 0.5$: $c(x) = 0$

$$\Rightarrow \boxed{\text{Bayes classifier}} = \begin{cases} 1 & P(Y=1|X=x) \geq 0.5 \\ 0 & P(Y=1|X=x) < 0.5 \end{cases}$$



Another confusion matrix

Researchers fit a model for the dengue data and produce the following confusion matrix:

Problem: Accuracy is

misleading with
imbalanced data

		Observed	
		$Y = 0$	$Y = 1$
Predicted	$\hat{Y} = 0$	3957	1631
	$\hat{Y} = 1$	66	66

good! bad!

The accuracy is 70%. Is the model doing a good job?

Sensitivity : $P(\hat{Y}=1 | Y=1)$ $\frac{66}{1631 + 66}$ (very low)

Specificity : $P(\hat{Y}=0 | Y=0)$ $\frac{3957}{3957 + 66}$ (very high)

Changing the threshold

Threshold of 0.3:

```
1 table("Predicted" = ifelse(m1$fitted.values > 0.3, 1, 0),  
2     "Observed" = m1$y)
```

Observed		Sens.	
Predicted	0	1	
0	301	46	
1	123	244	
		$\frac{244}{244+46} = 0.84$	
		Spec.	
		$\frac{301}{301+123} = 0.71$	

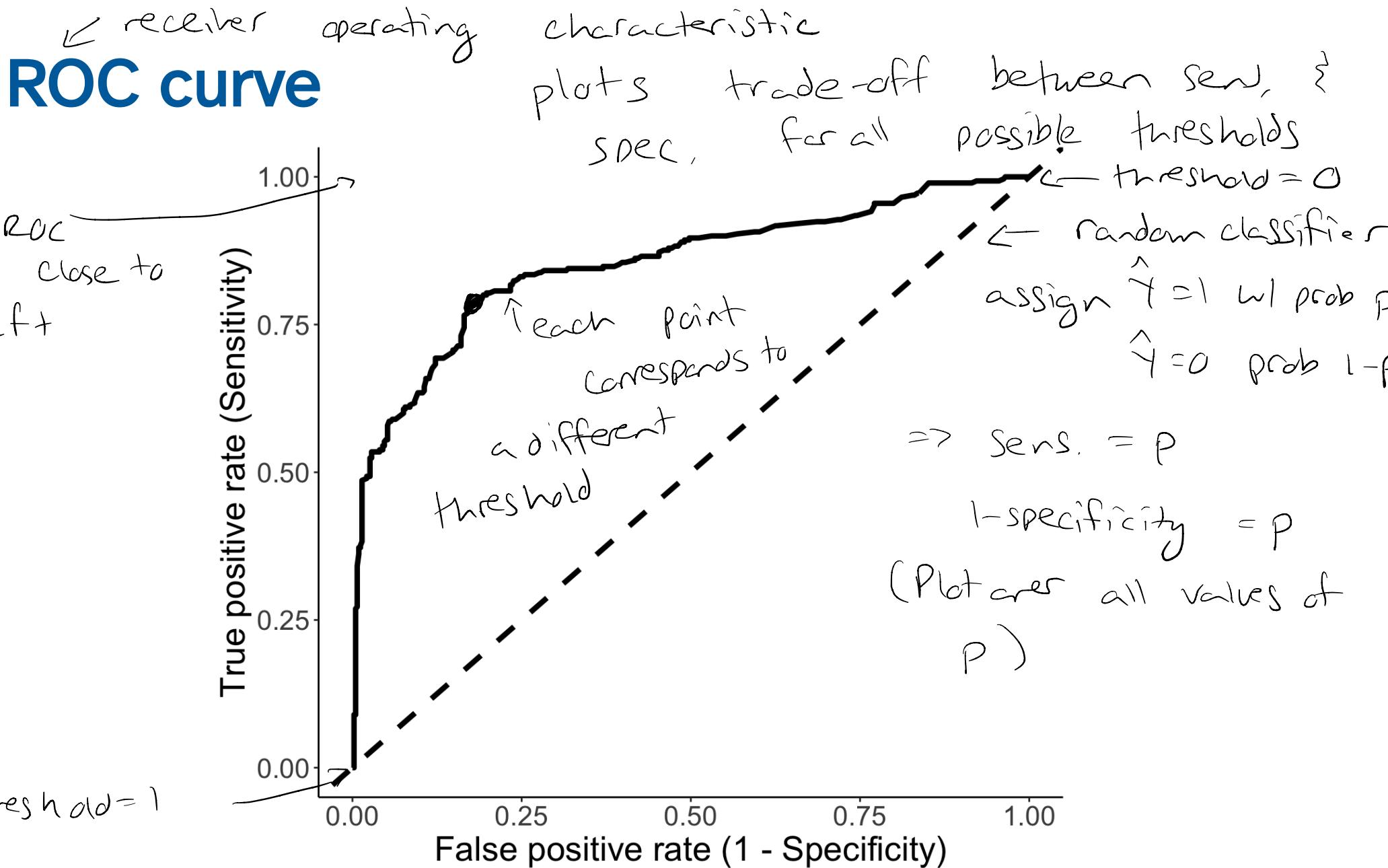
Threshold of 0.7:

```
1 table("Predicted" = ifelse(m1$fitted.values > 0.7, 1, 0),  
2     "Observed" = m1$y)
```

Observed		Sens.	
Predicted	0	1	
0	407	134	
1	17	156	
		Spec.	0.96

How do sensitivity and specificity change?

As threshold ↑, sensitivity ↓ and specificity ↑



Summarizing : area under curve (AUC)
 $0.5 \leq \text{AUC} \leq 1$ (closer to 1 is better)

Summary

- Threshold predicted probabilities to get binary predictions
- Performance metrics like accuracy, sensitivity, and specificity can be calculated from a confusion matrix
- A threshold of 0.5 maximizes accuracy (in the population)
- As threshold increases, sensitivity decreases and specificity increases
- ROC curves plot the trade-off between sensitivity and specificity

Class activity

- Take some time to work through the class activity
- You are welcome to work in groups

https://sta712-f23.github.io/class_activities/ca_lecture_3.html

Class activity: dengue data

```
1 m1 <- glm(Dengue ~ Age + WBC + PLT, data = dengue,  
2             family = binomial)  
3 m2 <- glm(Dengue ~ Age + WBC + PLT + BMI + HCT + Temperature,  
4             data = dengue, family = binomial)
```

How do I perform a LRT to compare the two models?

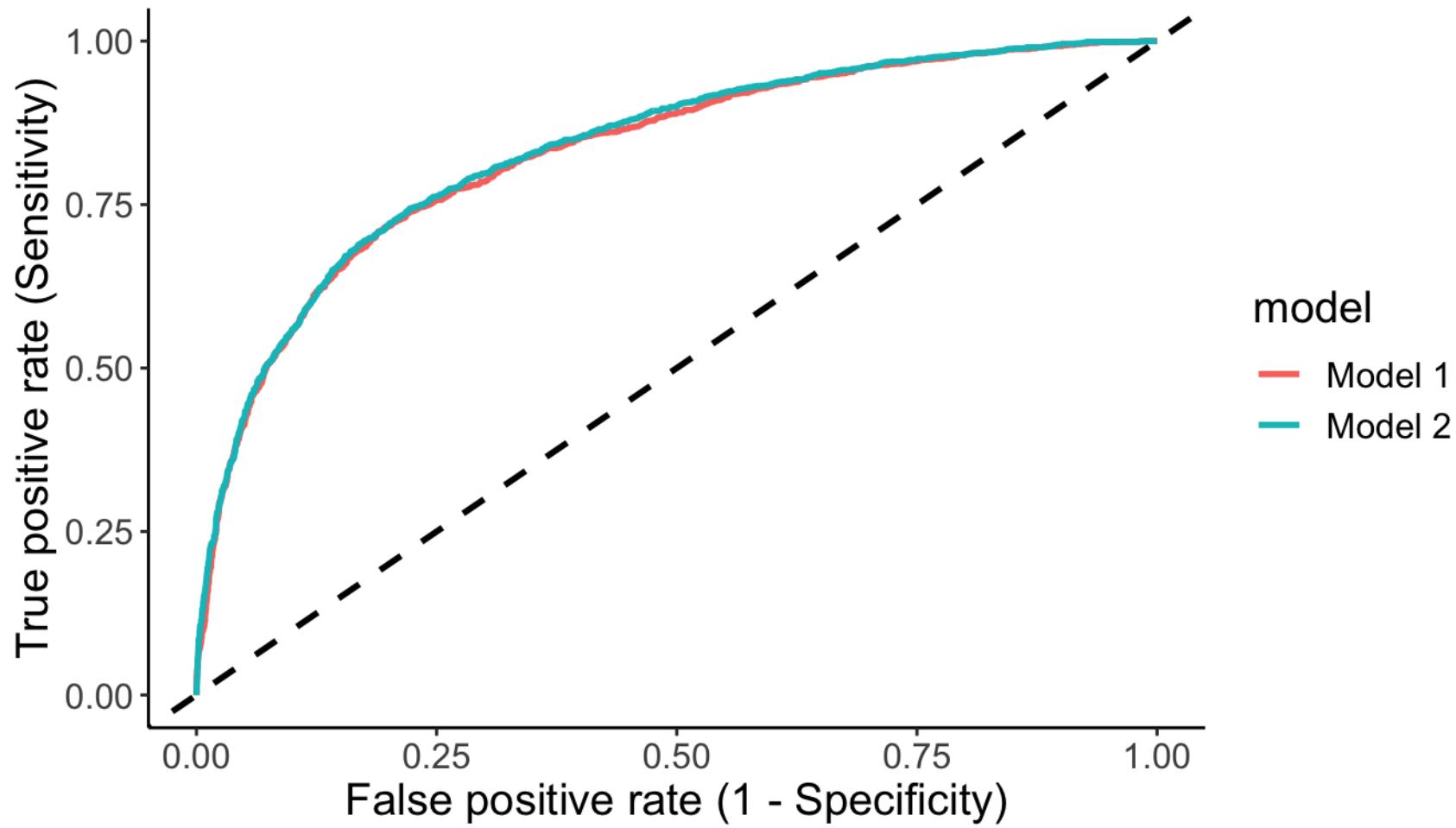
Class activity: dengue data

```
1 m1 <- glm(Dengue ~ Age + WBC + PLT, data = dengue,  
2             family = binomial)  
3 m2 <- glm(Dengue ~ Age + WBC + PLT + BMI + HCT + Temperature,  
4             data = dengue, family = binomial)  
5  
6 pchisq(m1$deviance - m2$deviance, 3, lower.tail=F)
```

[1] 1.742259e-16

Which model would the LRT choose?

Class activity: dengue data



Which model would you choose?

Class activity: simulated data

```
1 m1 <- glm(y ~ X[,1:3], family = binomial)
2 m2 <- glm(y ~ X, family = binomial)
3
4 m1$deviance
```

```
[1] 1294.233
```

```
1 m2$deviance
```

```
[1] 1292.818
```

Why *must* the second model have a smaller deviance (on the data used to fit the model)?

Class activity: simulated data

```
1 pred1 <- prediction(m1$fitted.values, m1$y)
2 pred2 <- prediction(m2$fitted.values, m2$y)
3
4 performance(pred1, "auc")@y.values
```

```
[[1]]
[1] 0.5808683
```

```
1 performance(pred2, "auc")@y.values
```

```
[[1]]
[1] 0.5847598
```

Why might model 2 have a greater AUC?

Class activity: new simulated data

```
1 # predictions on new observations
2 phat_m1 <- exp(m1$coefficients[1] + X_new[,1:3] %*% m1$coefficients[2])
3   + exp(m1$coefficients[1] + X_new[,1:3] %*% m1$coefficients[2:4])
4 )
5
6 phat_m2 <- exp(m2$coefficients[1] + X_new %*% m2$coefficients[2:7])/
7   1 + exp(m2$coefficients[1] + X_new %*% m2$coefficients[2:7])
8 )
9
10 # new deviance for model 1
11 -2*sum(y_new*log(phat_m1) + (1-y_new)*log(1 - phat_m1))
```

```
[1] 1321.424
```

```
1 # new deviance for model 2
2 -2*sum(y_new*log(phat_m2) + (1-y_new)*log(1 - phat_m2))
```

```
[1] 1321.842
```

Why are the deviances higher than for the training data?

Key take-aways

- When evaluated on the *training* data, a large model will have a lower deviance than any of its sub-models
 - Prediction metrics like AUC are also often higher, even if a reduced model is correct
- Often prefer the simpler model (easier to interpret, less variability, etc.) if model performance is similar, *even if* a hypothesis test would choose the larger model
- We expect model performance (deviance, AUC, etc.) to be better on training data than on a new test set

Training vs. testing data

- Models generally perform better on their *training* data (the data used to fit the model) than on new (*test*) data.
- When evaluated only on training data, larger models tend to look better

How should we assess and compare model performance if we can't sample new data (e.g., in the dengue scenario)?

Data splitting

How should we assess and compare model performance if we can't sample new data (e.g., in the dengue scenario)?

- Randomly divide available data into two groups: training and test
 - E.g. 70% training, 30% test
- Fit the model on the training sample
- Evaluate the model on the test sample

Data splitting with the dengue data

```
1 # create training and test splits  
2 train_sample <- sample(1:nrow(dengue), 0.7*nrow(dengue), replace = F)  
3 dengue_train <- dengue[train_sample,]  
4 dengue_test <- dengue[setdiff(1:nrow(dengue), train_sample),]  
  
1 # fit the model  
2 m1_train <- glm(Dengue ~ Age + WBC + PLT, data = dengue_train,  
3                   family = binomial)  
4  
5 # predict on test data  
6 test_predictions <- predict(m1_train, newdata = dengue_test,  
7                               type = "response")  
8 pred <- prediction(test_predictions, dengue_test$Dengue)  
9 performance(pred, "auc")@y.values  
  
[[1]]  
[1] 0.8252114
```

Are there any potential issues with this strategy?

Downsides of train/test splits

- We get less data for training
- Performance measure depends on the (random) split

Alternative: cross-validation

Cross validation

- Divide data into k groups (*folds*)
- For each fold $i = 1, \dots, k$:
 - Train model on the remaining $k - 1$ folds
 - Evaluate on fold i
- Average performance across the k folds

Key take-aways

- Don't choose a model based solely on training performance
 - Will bias towards more complex models
- Train/test splits and cross-validation give better estimates of model performance

