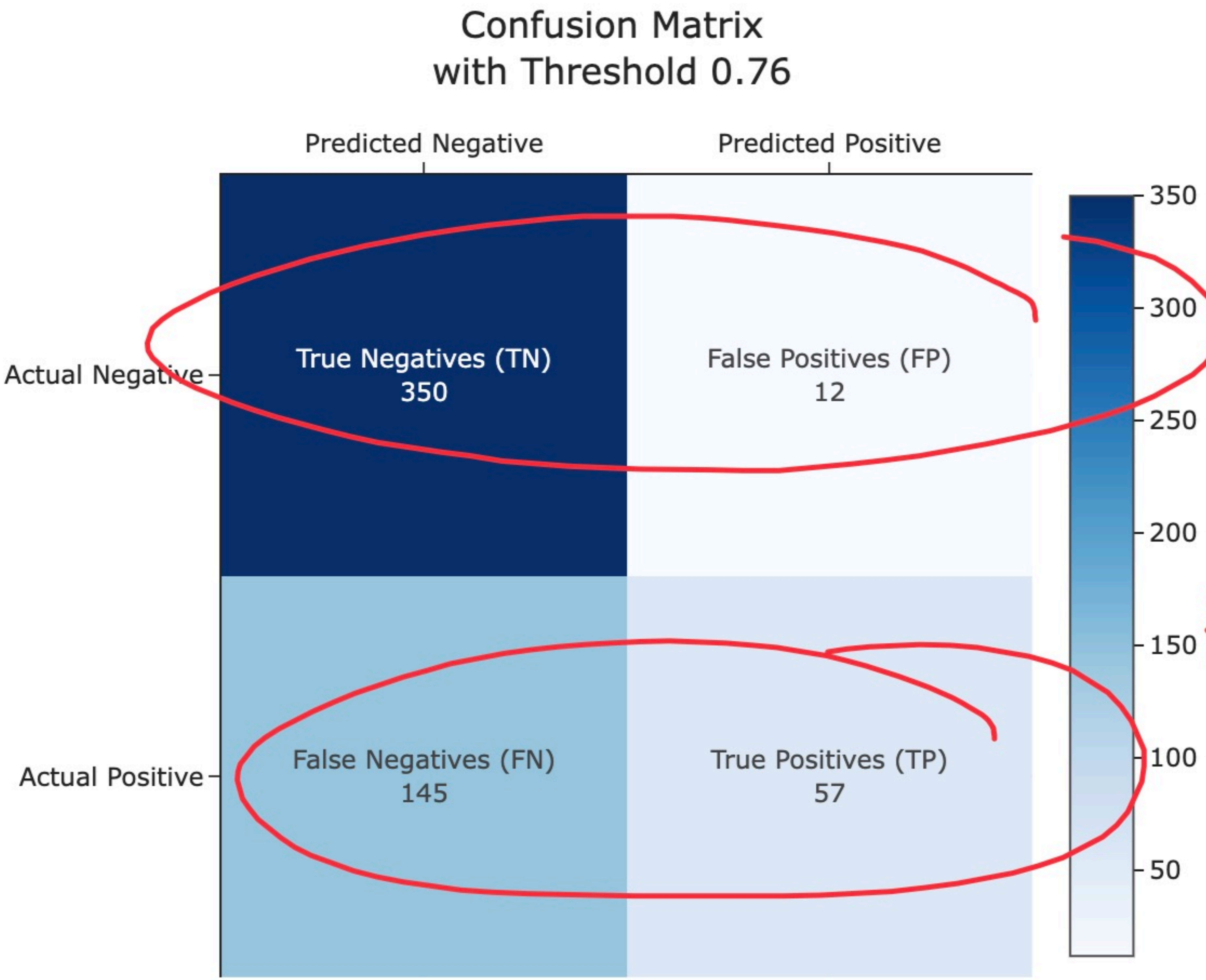
localhost

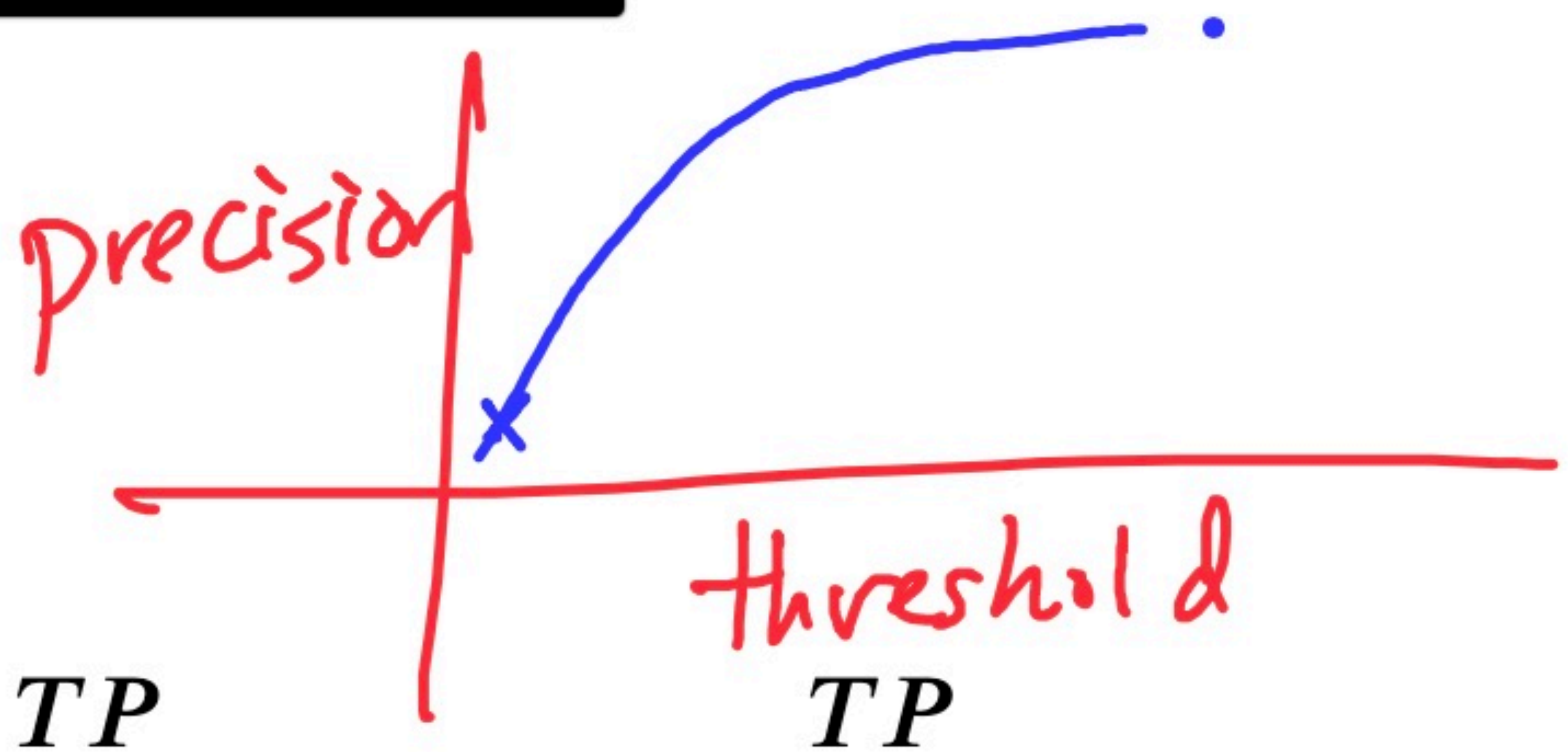
T 0.76



TN+FP is constant (362)

FN+TP is constant (202).

Precision vs. threshold



- Precision is defined as:

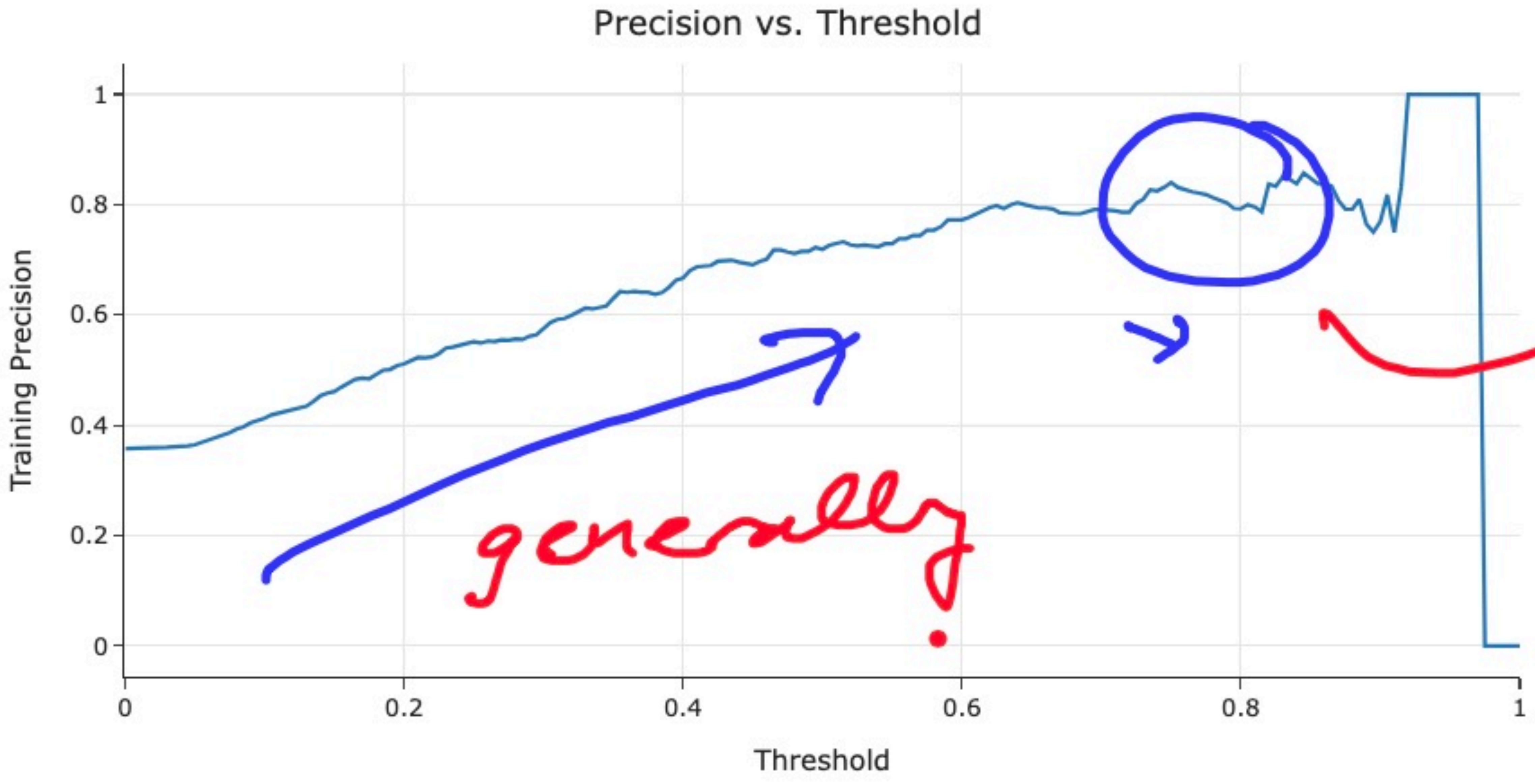
$$\text{precision} = \frac{TP}{\#\text{ predicted positive}} = \frac{TP}{TP + FP}$$

Here, a false positive (FP) is when we predict that someone has diabetes when they do not.

- How does the model's training **precision** change as the threshold changes?

In [ ]:
```
util.plot_vs_threshold(X_train, y_train, 'Precision')
```

In [22]:
```python
util.plot_vs_threshold(X_train, y_train, 'Precision')
```



rare case

can go down if, e.g.

$$\frac{3}{3+1} \rightarrow \frac{2}{2+1}$$

$3/4$ $\qquad$ $2/3$

- If the "bar" is higher to predict 1, then we will have fewer positives in general, and thus fewer false positives.

- As the **threshold increases** ⬆️, the denominator in precision $= \frac{TP}{TP+FP}$ will decrease, and so **precision tends to increase** ⬆️.

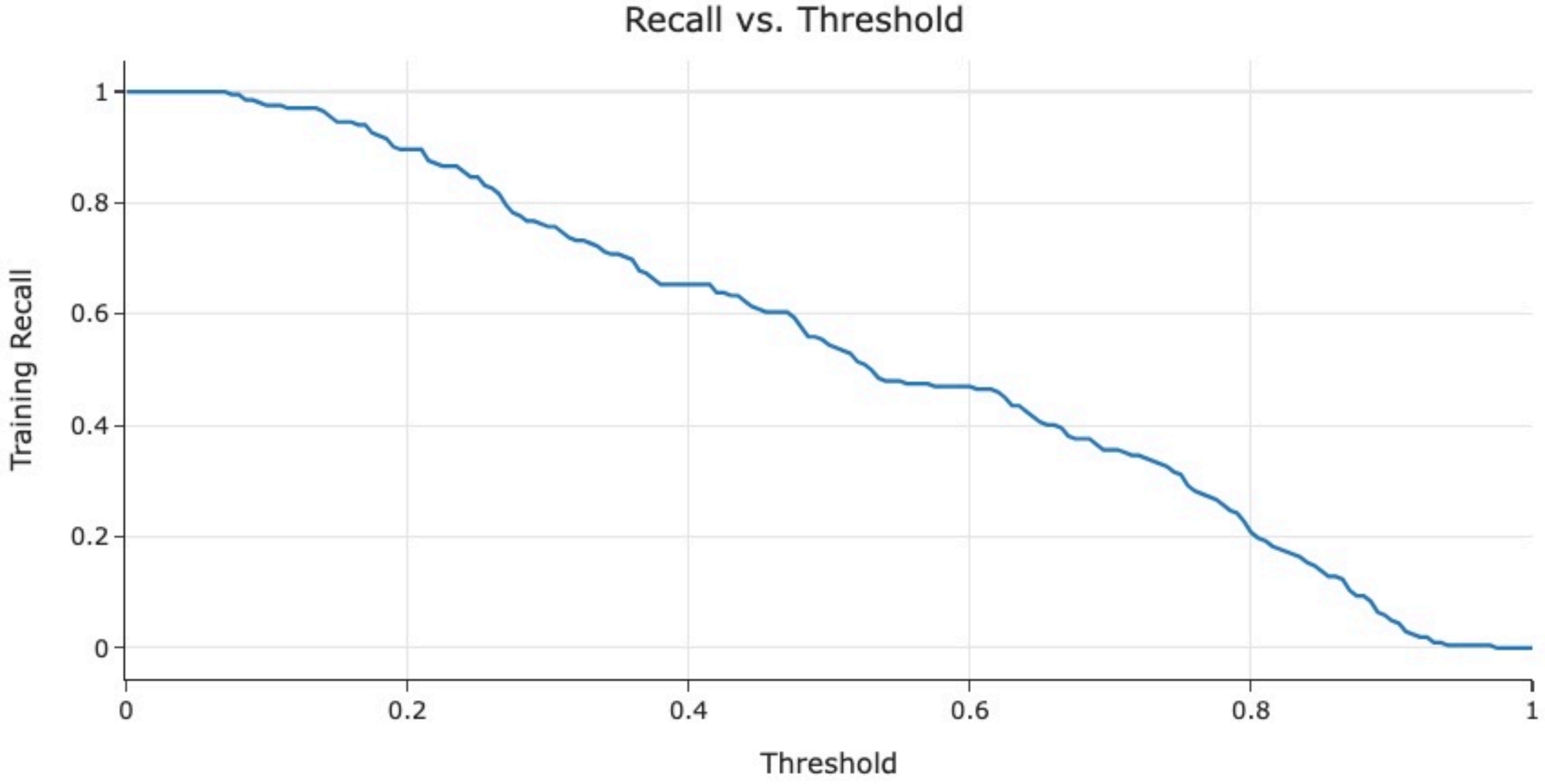
There are some cases where a slightly higher threshold led to a slightly lower precision; why?

$$\text{recall} = \frac{TP}{\text{\# actually positive}} = \frac{TP}{TP + FN}$$ ← constant.

Here, a false negative (FN) is when we predict that someone does not have diabetes, when they really do.

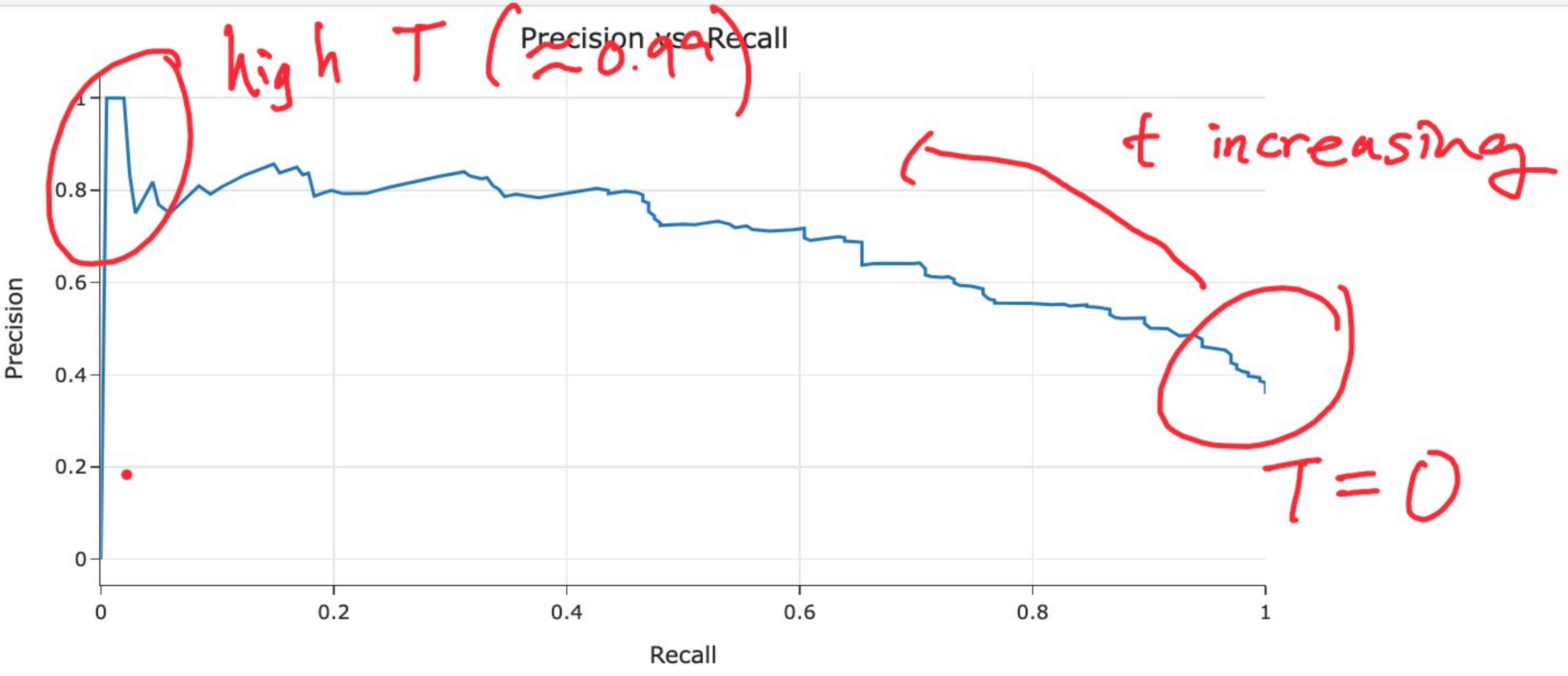
- How does the model's training **recall** change as the threshold changes?

In [23]:
```
util.plot_vs_threshold(X_train, y_train, 'Recall')
```



- We can visualize how precision and recall vary **together**.

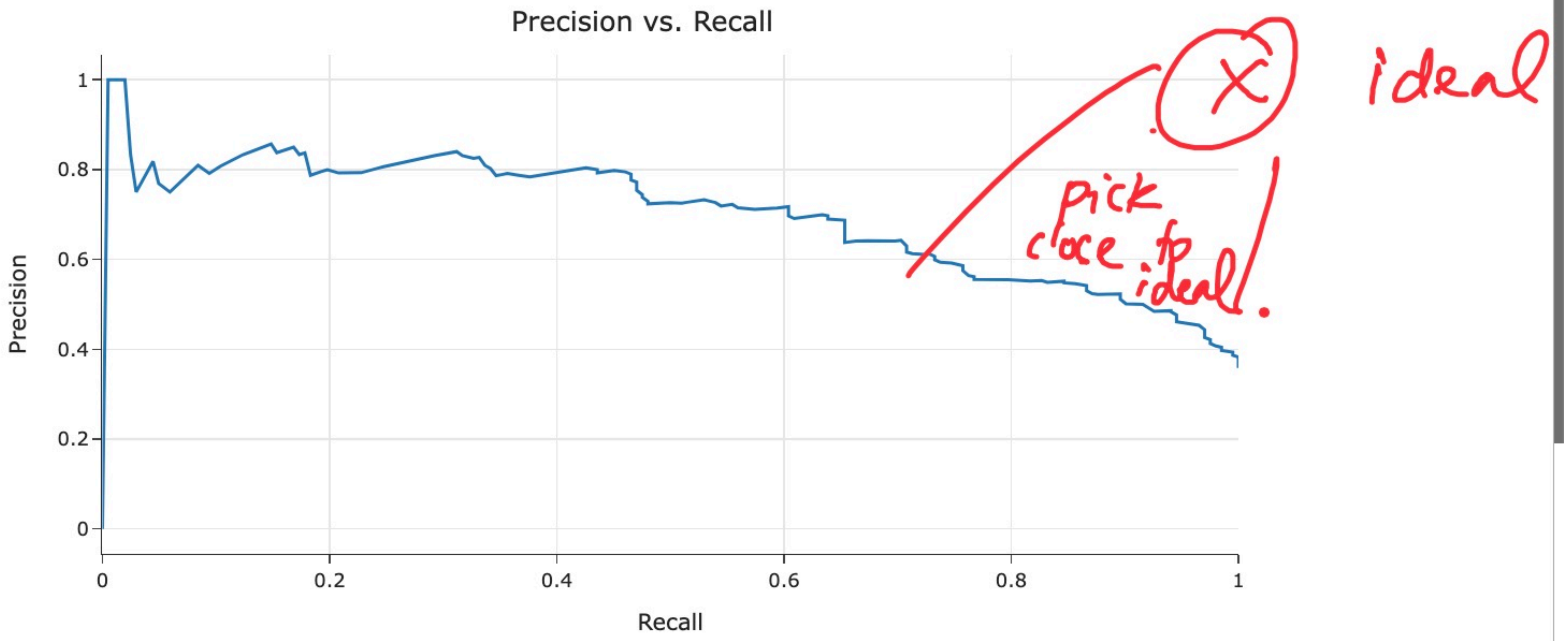
In [24]:
```
util.pr_curve(X_train, y_train)
```



- The curve above is called a **PR curve**.

- We can visualize how precision and recall vary **together**.

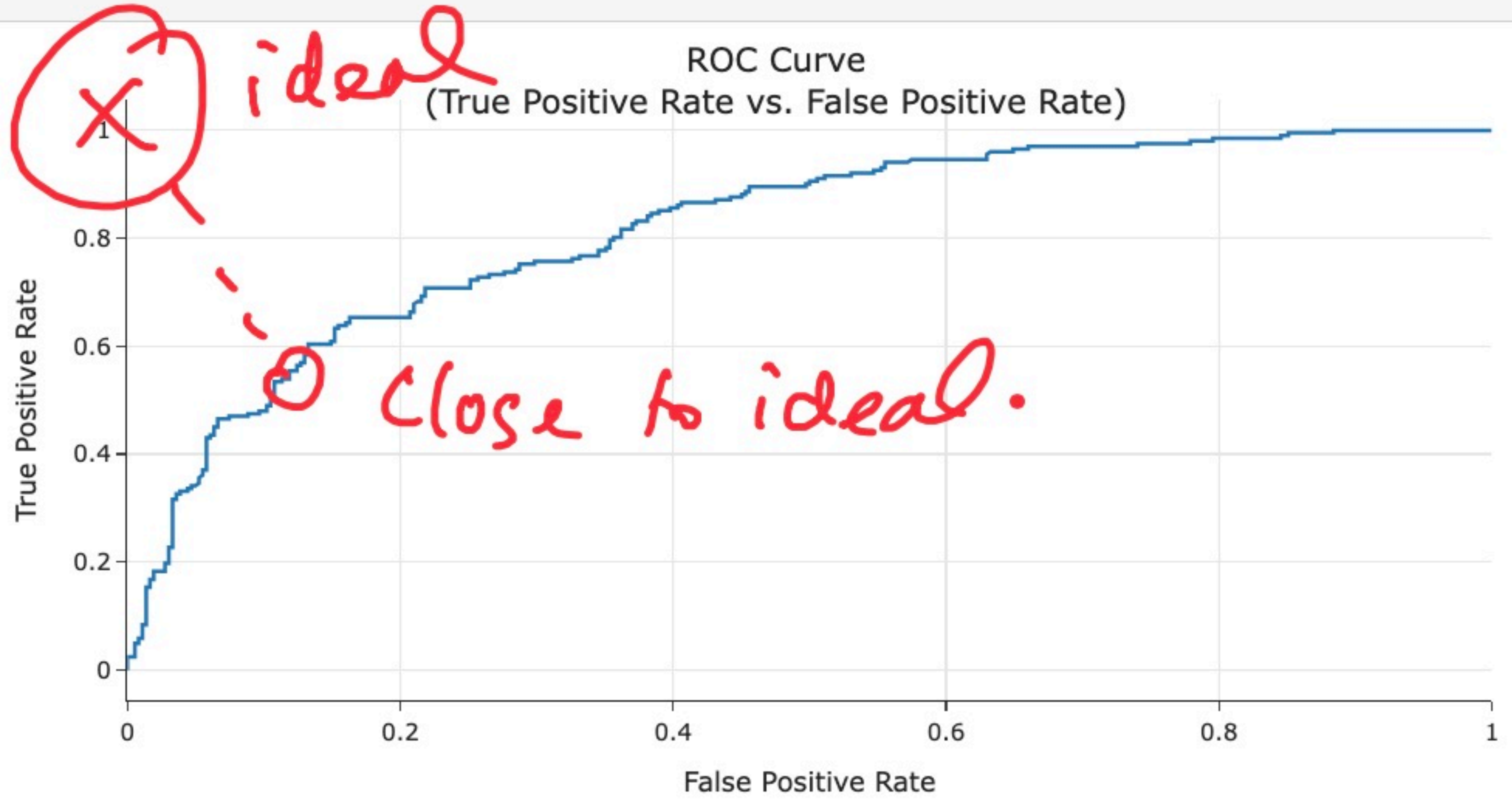
```
In [24]:  util.pr_curve(X_train, y_train)
```



- The curve above is called a **PR curve**.

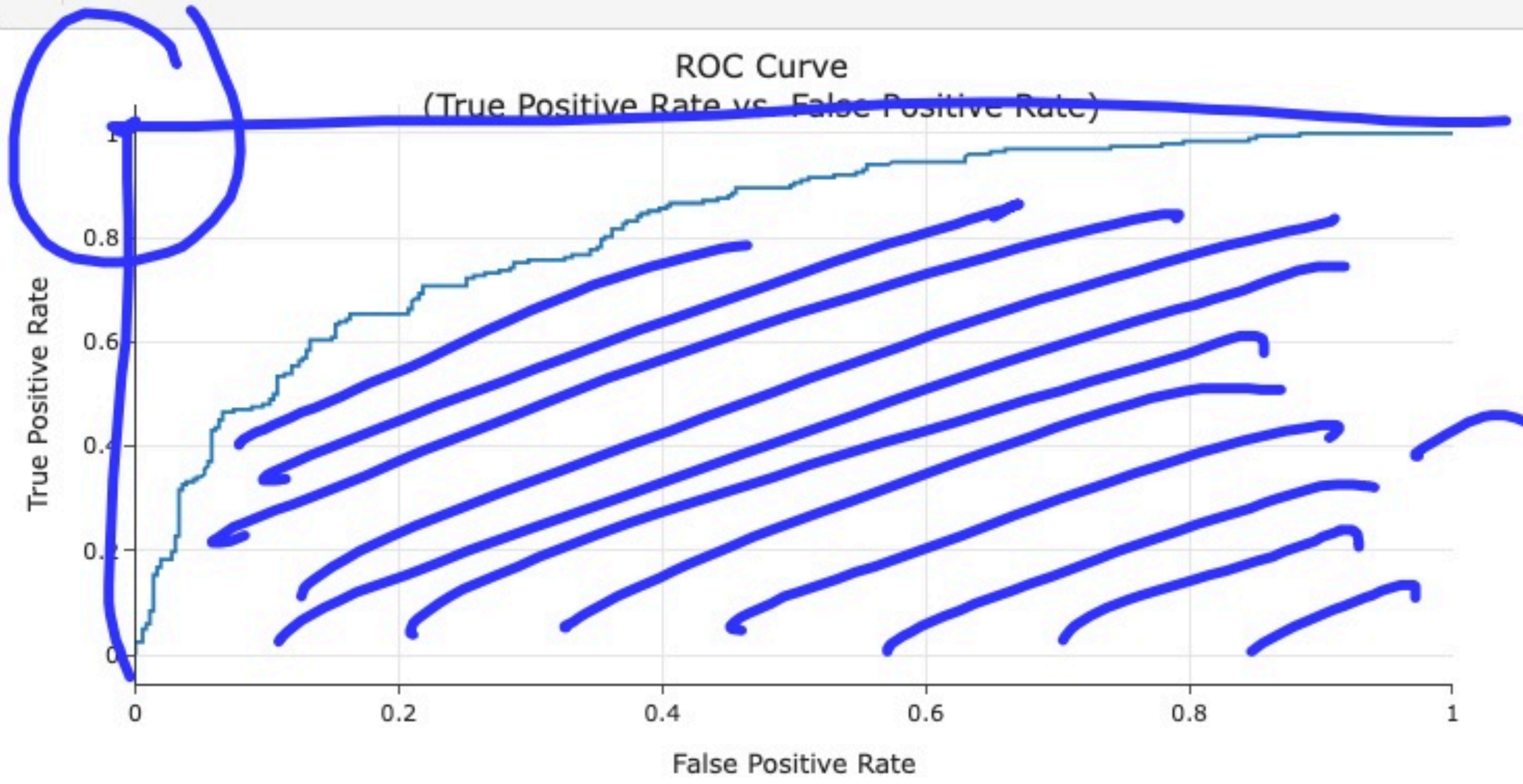
The ROC curve for our classifier looks like:

In [25]:
```
util.draw_roc_curve(X_train, y_train)
```



The ROC curve for our classifier looks like:

In [25]:
```
util.draw_roc_curve(X_train, y_train)
```



AUC : want close to 1.

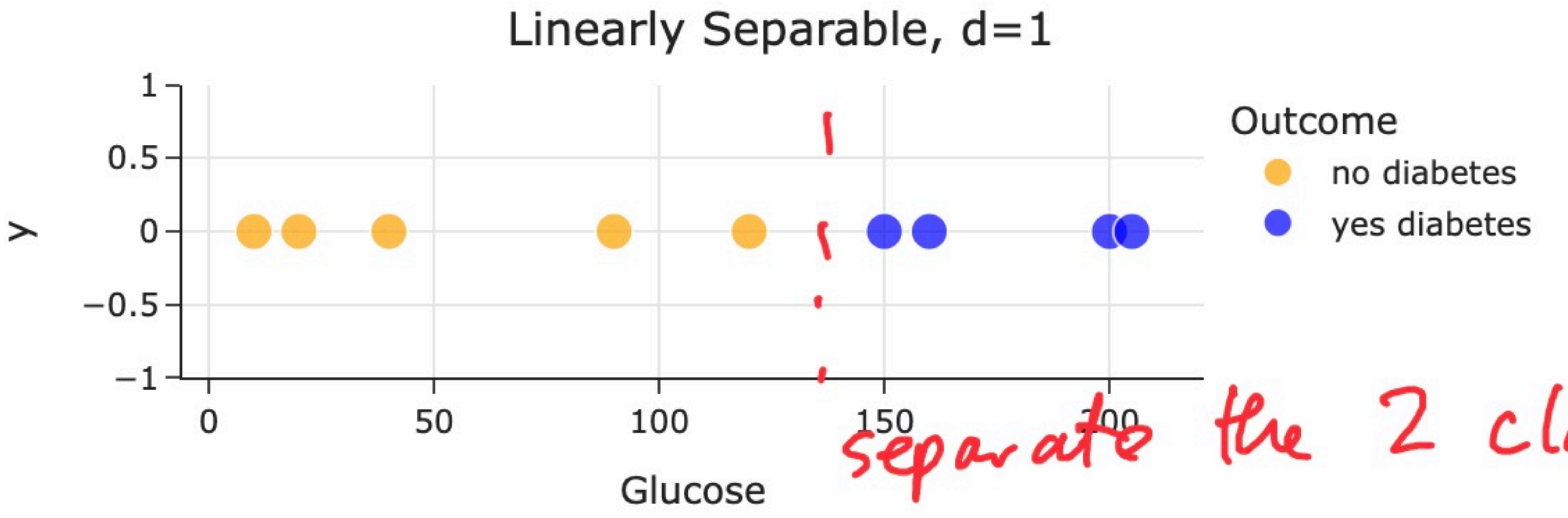
- If we care about TPR and FPR equally, the best threshold is the one whose point is closest to the **top left corner** in the plot above.

  Why? The top left corner is where $TPR = 1$ and $FPR = 0$, and we want TPR to be high and FPR to be low.

- A common metric for the quality of a binary classifier is the **area under curve (AUC)** for the ROC curve.

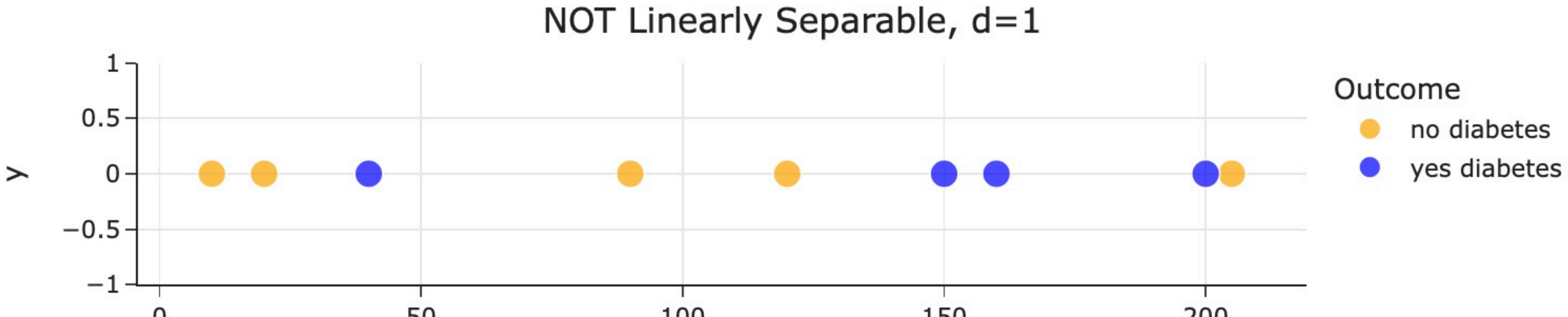
  Larger values are better!

```
In [29]: 1 util.lin_sep_1D()
```



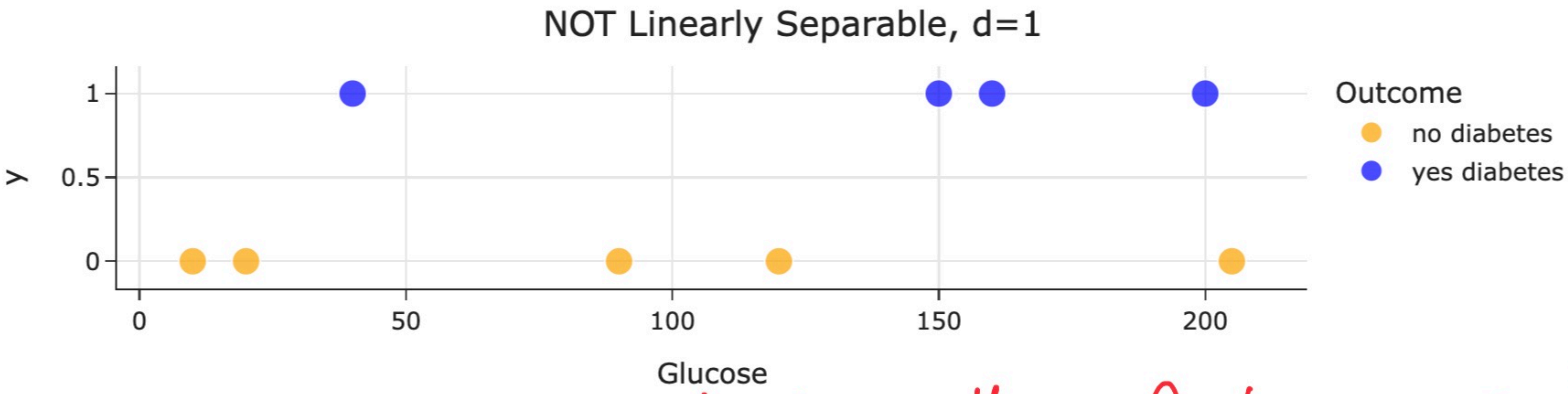
separate the 2 classes!

```
In [28]: 1 util.non_lin_sep_1D()
```



[32]:

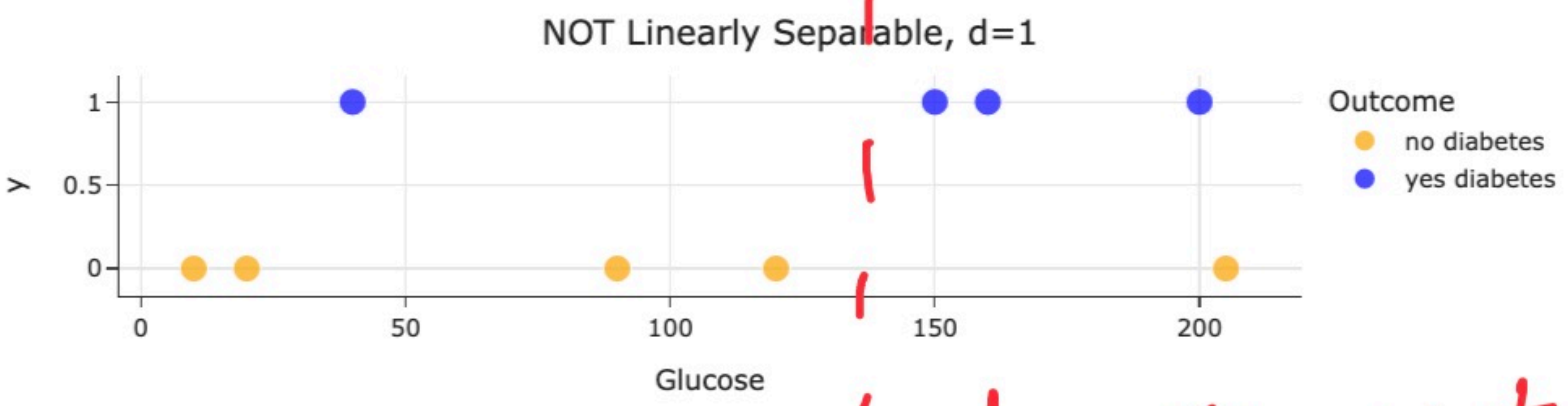
```
1 util.bad_example_1D()
```



not in the feature space!

- Why is the dataset below **not** linearly separable?

In [32]:
```
1 util.bad_example_1D()
```



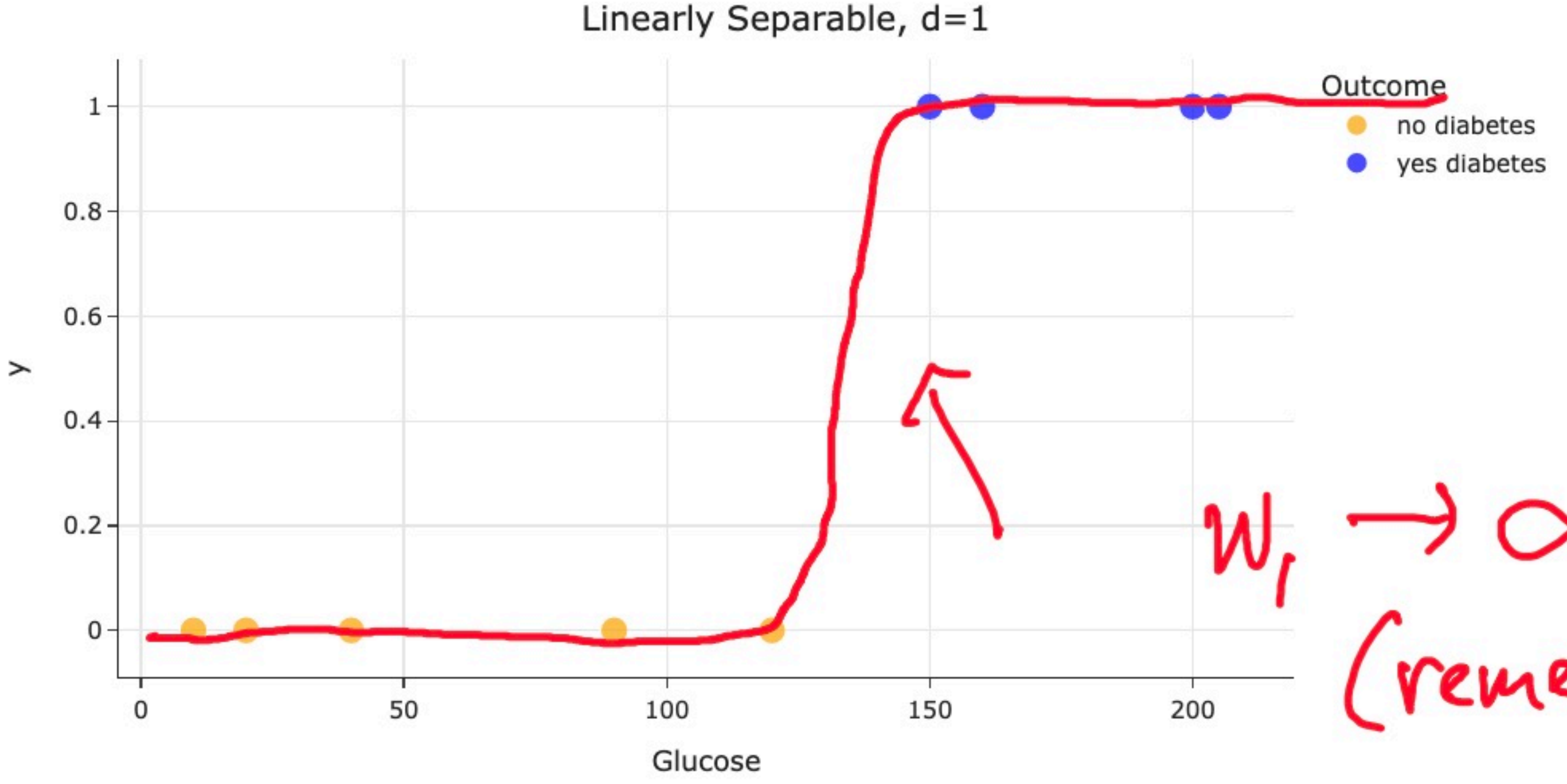
| doesn't exist.

See the annotated slides for more details.

$$P(y_i = 1|\text{Glucose}_i) = \sigma(w_0 + w_1 \cdot \text{Glucose}_i) = \frac{1}{1 + e^{-(w_0 + w_1 \cdot \text{Glucose}_i)}}$$

In [35]:

```
util.lin_sep_1D_elevated()
```



$w_1 \rightarrow \infty$

(remember, w_1 controls steepness)

28 . 1

cover rest
on Tuesday →.

Logistic regression for multiclass classification