
Univ.AI

Univ.AI

Classification

Logistic Regression

Univ.AI

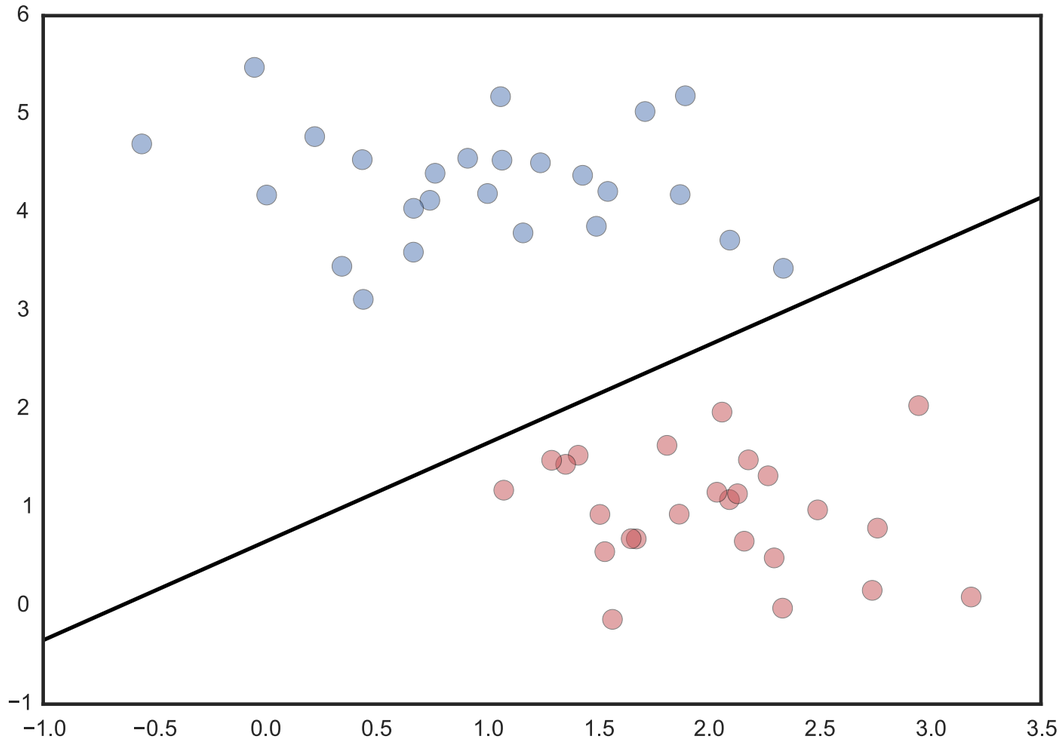
Last time

- Validation
- Cross Validation
- Regularization
- You were supposed to do the regularization part of the lab

This time

1. Multiple Regression and Dimensionality
2. Classification
3. Logistic Regression
4. Metrics
5. Predictive distributions (back to regression)
6. Bayes Risk (for regression)
7. Bayes Risk for Classification

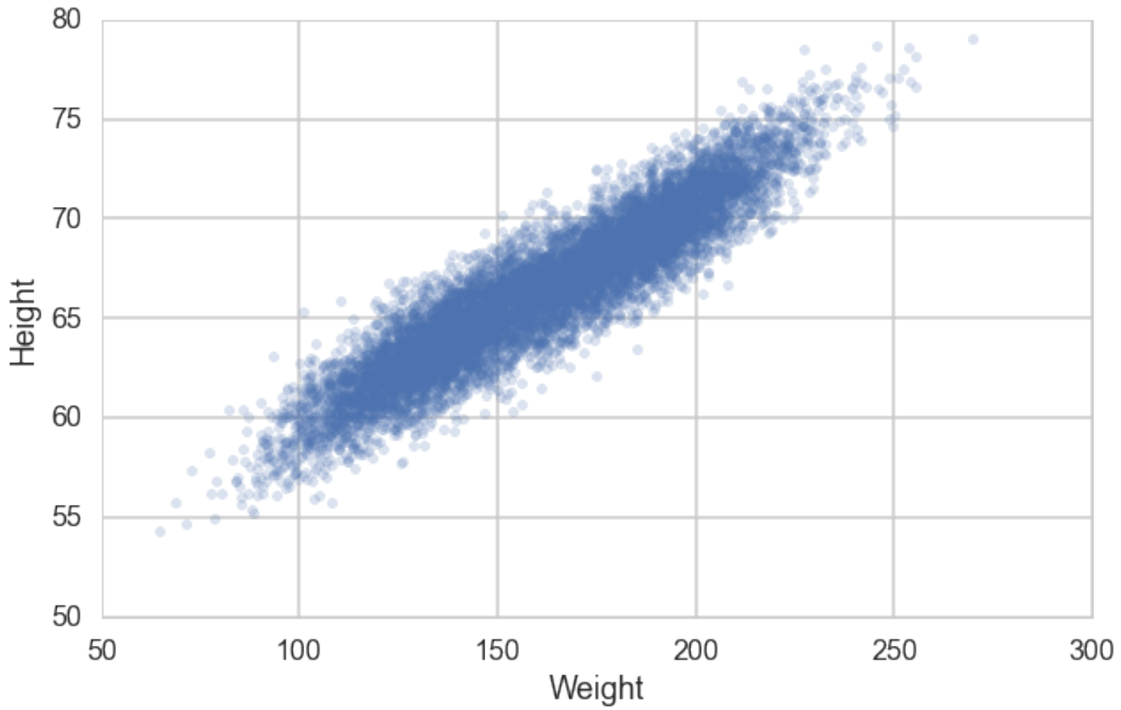
2. CLASSIFICATION

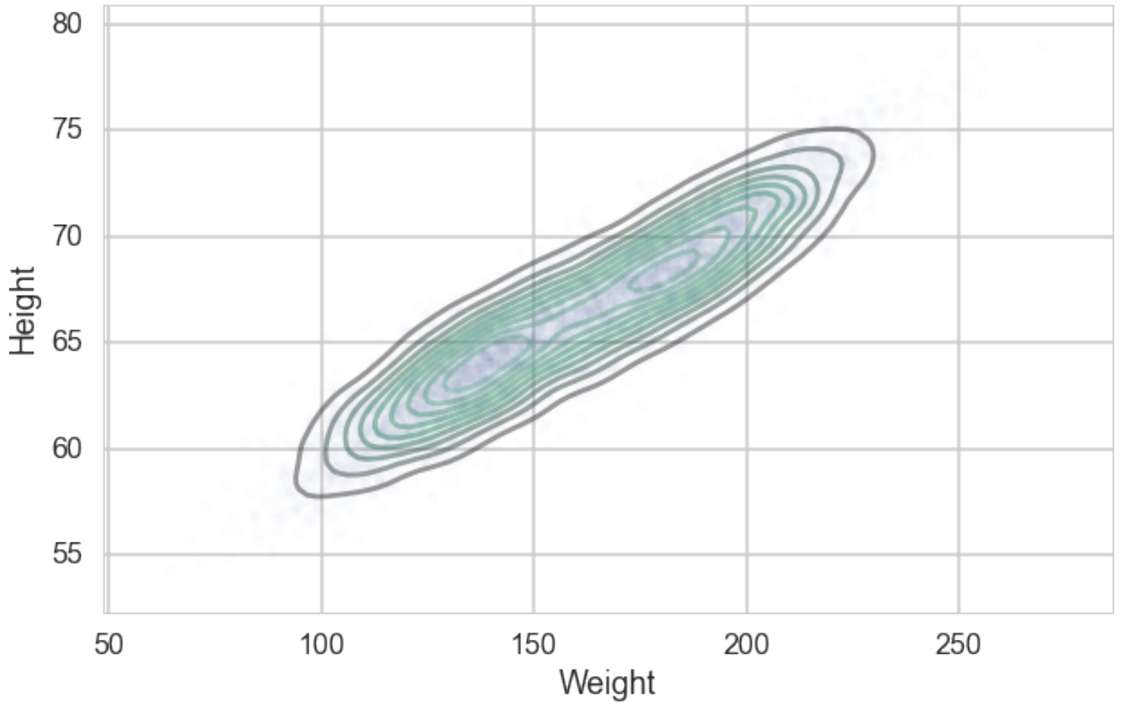


- will a customer churn?
- is this a check? For how much?
- a man or a woman?
- will this customer buy?
- do you have cancer?
- is this spam?
- whose picture is this?
- what is this text about?[j]

[j] image from code in http://bit.ly/1Azg29G

PROBABILISTIC CLASSIFICATION





In any machine learning problem we want to model $p(x, y)$.

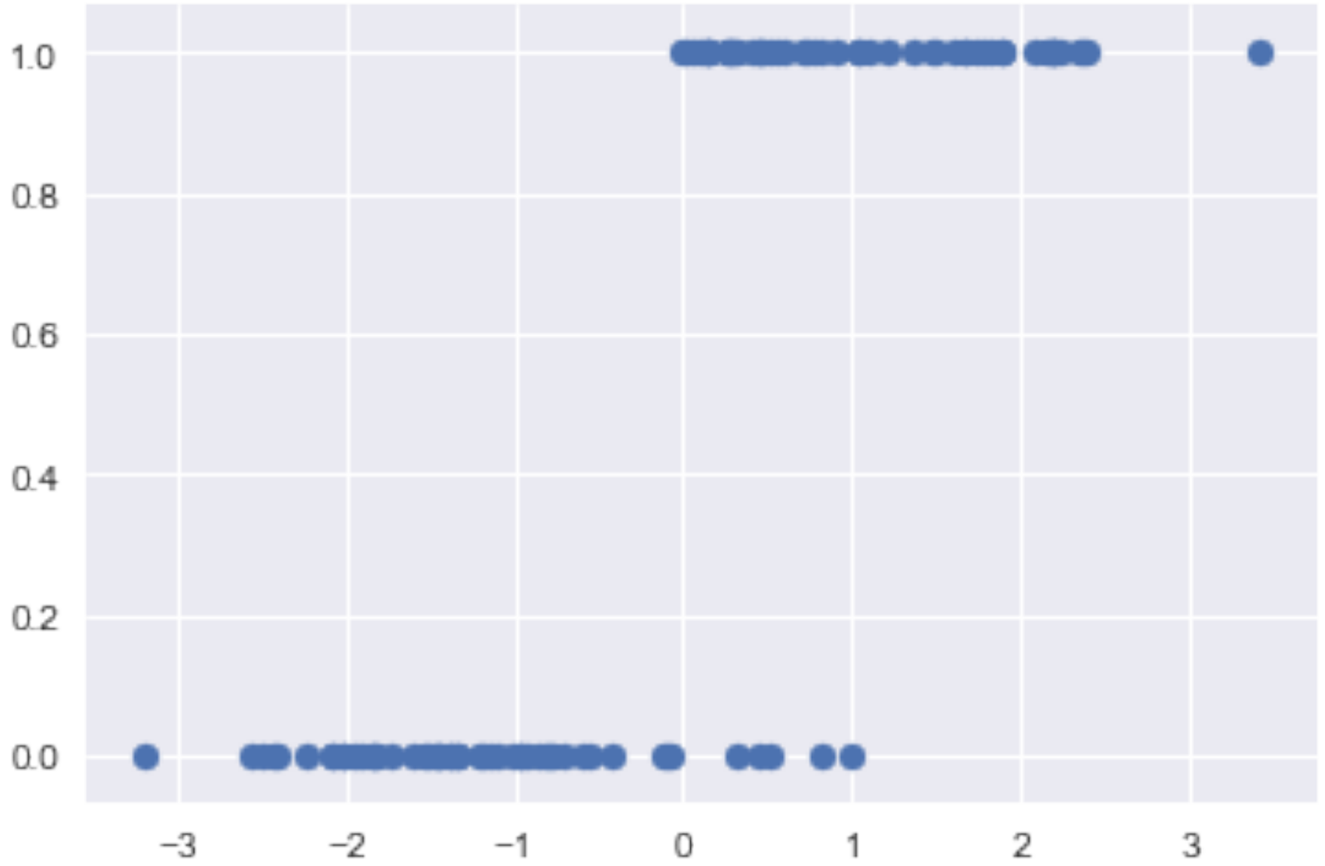
We can choose to model as

$p(x, y) = p(y \mid x)p(x)$ or $p(x \mid y)p(y)$

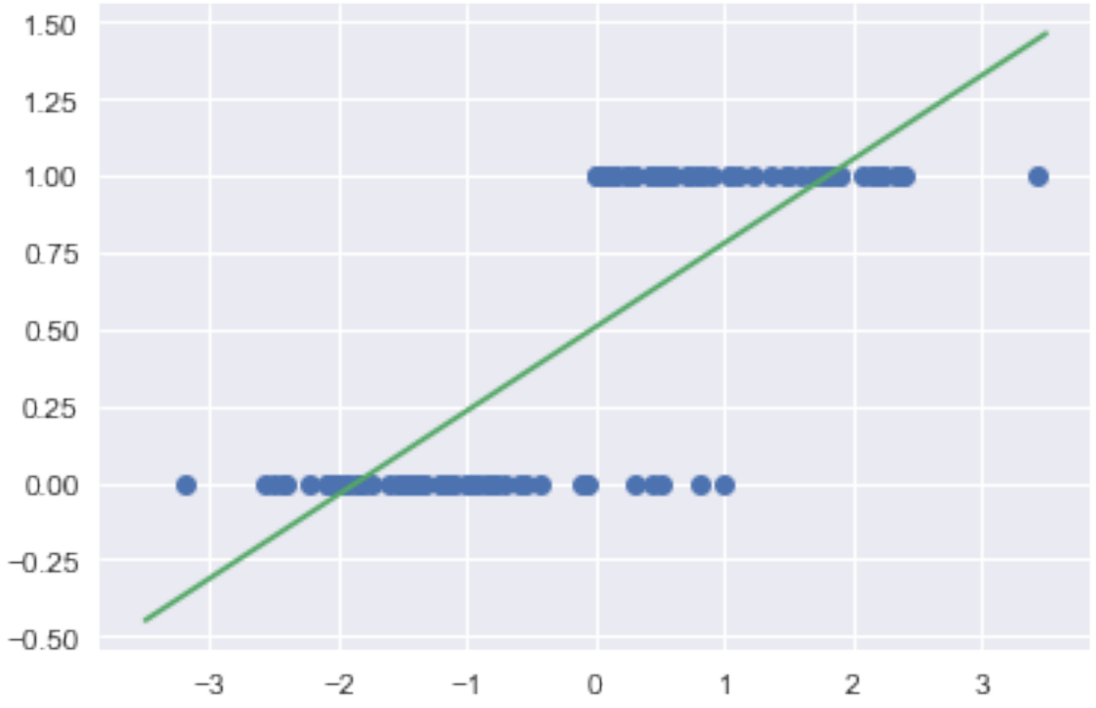
In regression we modeled the former. In logistic regression, with $y = c$ (class c) we model the former as well. This is the probability of the class given the features x.

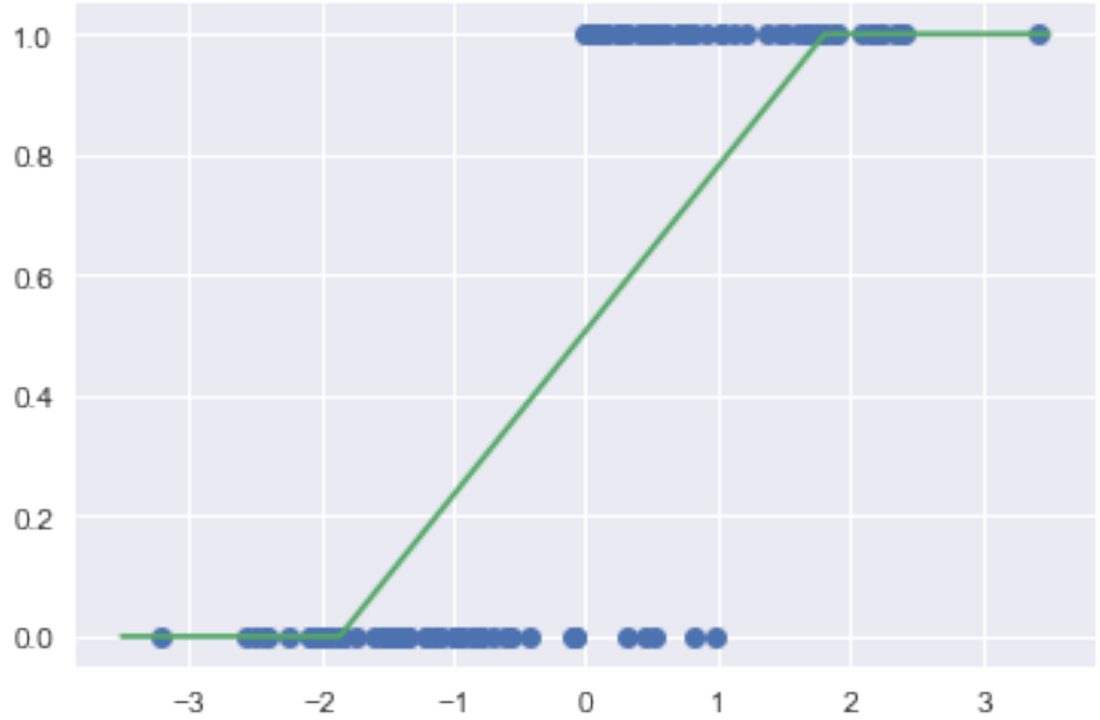
In "Generative models" we model the latter, the probability of the features given the class.

1-D classification problem



1-D Using Linear regression





3. MLE for Logistic Regression

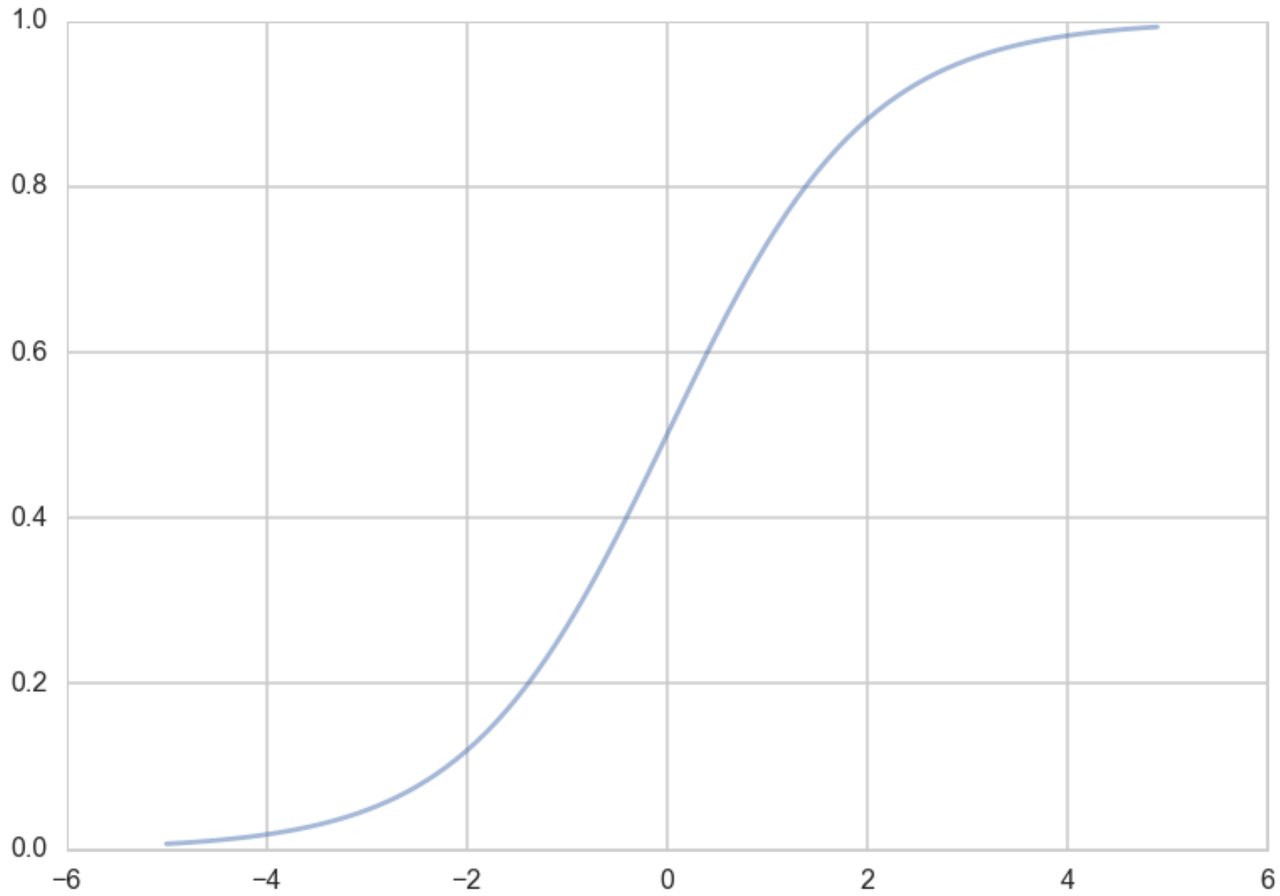
- example of a Generalized Linear Model (GLM)
- "Squeeze" linear regression through a **Sigmoid** function
- this bounds the output to be a probability
- What is the sampling Distribution?

Sigmoid function

This function is plotted below:

```
h = lambda z: 1./(1+np.exp(-z))
zs=np.arange(-5,5,0.1)
plt.plot(zs, h(zs), alpha=0.5);
```

Identify: $z = \mathbf{w} \cdot \mathbf{x}$ and $h(\mathbf{w} \cdot \mathbf{x})$ with the probability that the sample is a '1' ($y = 1$).



Then, the conditional probabilities of $y = 1$ or $y = 0$ given a particular sample's features $\mathbf{x}$ are:

$$P(y = 1|\mathbf{x}) = h(\mathbf{w} \cdot \mathbf{x})$$
$$P(y = 0|\mathbf{x}) = 1 - h(\mathbf{w} \cdot \mathbf{x}).$$

These two can be written together as

$$P(y|\mathbf{x}, \mathbf{w}) = h(\mathbf{w} \cdot \mathbf{x})^y (1 - h(\mathbf{w} \cdot \mathbf{x}))^{(1-y)}$$

BERNOULLI!!

$$\ln\left(\frac{P(y=1|\mathbf{x})}{1-P(y=1|\mathbf{x})}\right) = \beta_0 + \beta_1 x$$

Logistic regression models the log-odds with a linear function of the predictors or features. Thus a one unit change in x is associated with a change in the odds of e^{β_1} .

Multiplying over the samples we get:

$$P(y|\mathbf{x}, \mathbf{w}) = P(\{y_i\}|\{\mathbf{x}_i\}, \mathbf{w}) = \prod_{y_i \in \mathcal{D}} P(y_i|\mathbf{x}_i, \mathbf{w}) = \prod_{y_i \in \mathcal{D}} h(\mathbf{w} \cdot \mathbf{x}_i)^{y_i} (1 - h(\mathbf{w} \cdot \mathbf{x}_i))^{(1-y_i)}$$

Indeed its important to realize that a particular sample can be thought of as a draw from some "true" probability distribution.

maximum likelihood estimation maximises the **likelihood of the sample y**, or alternately the log-likelihood,

$\mathcal{L} = P(y \mid \mathbf{x}, \mathbf{w})$. OR $\ell = log(P(y \mid \mathbf{x}, \mathbf{w}))$

Thus

$$
\begin{aligned}
\ell &= log\left(\prod_{y_i \in \mathcal{D}} h(\mathbf{w}\cdot\mathbf{x}_i)^{y_i}\left(1-h(\mathbf{w}\cdot\mathbf{x}_i)\right)^{(1-y_i)}\right) \\
&= \sum_{y_i \in \mathcal{D}} log\left(h(\mathbf{w}\cdot\mathbf{x}_i)^{y_i}\left(1-h(\mathbf{w}\cdot\mathbf{x}_i)\right)^{(1-y_i)}\right) \\
&= \sum_{y_i \in \mathcal{D}} log\, h(\mathbf{w}\cdot\mathbf{x}_i)^{y_i} + log\left(1-h(\mathbf{w}\cdot\mathbf{x}_i)\right)^{(1-y_i)} \\
&= \sum_{y_i \in \mathcal{D}} (y_i log(h(\mathbf{w}\cdot\mathbf{x})) + (1-y_i) log(1-h(\mathbf{w}\cdot\mathbf{x})))
\end{aligned}
$$

Logistic Regression: NLL

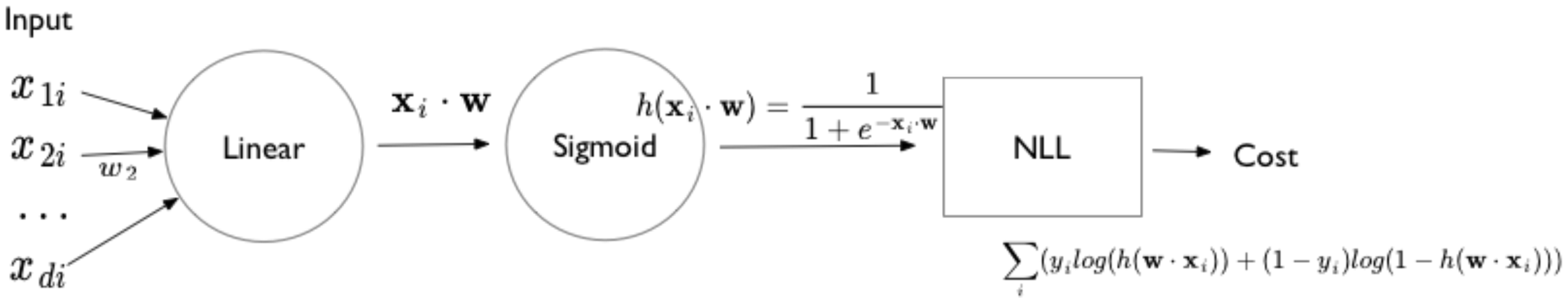
The negative of this log likelihood (NLL), also called *cross-entropy*.

$$NLL = -\sum_{y_i \in \mathcal{D}} (y_i log(h(\mathbf{w} \cdot \mathbf{x})) + (1 - y_i) log(1 - h(\mathbf{w} \cdot \mathbf{x})))$$

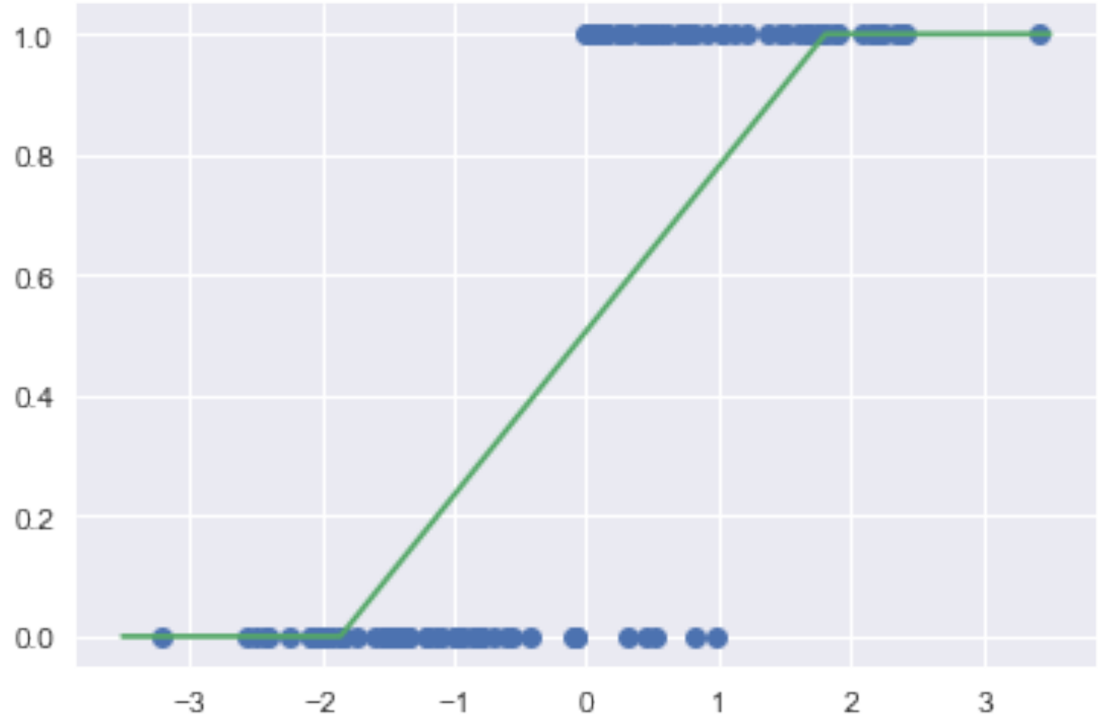
Gradient: $\nabla_{\mathbf{w}} NLL = \sum_i \mathbf{x}_i^T (p_i - y_i) = \mathbf{X}^T \cdot (\mathbf{p} - \mathbf{w})$

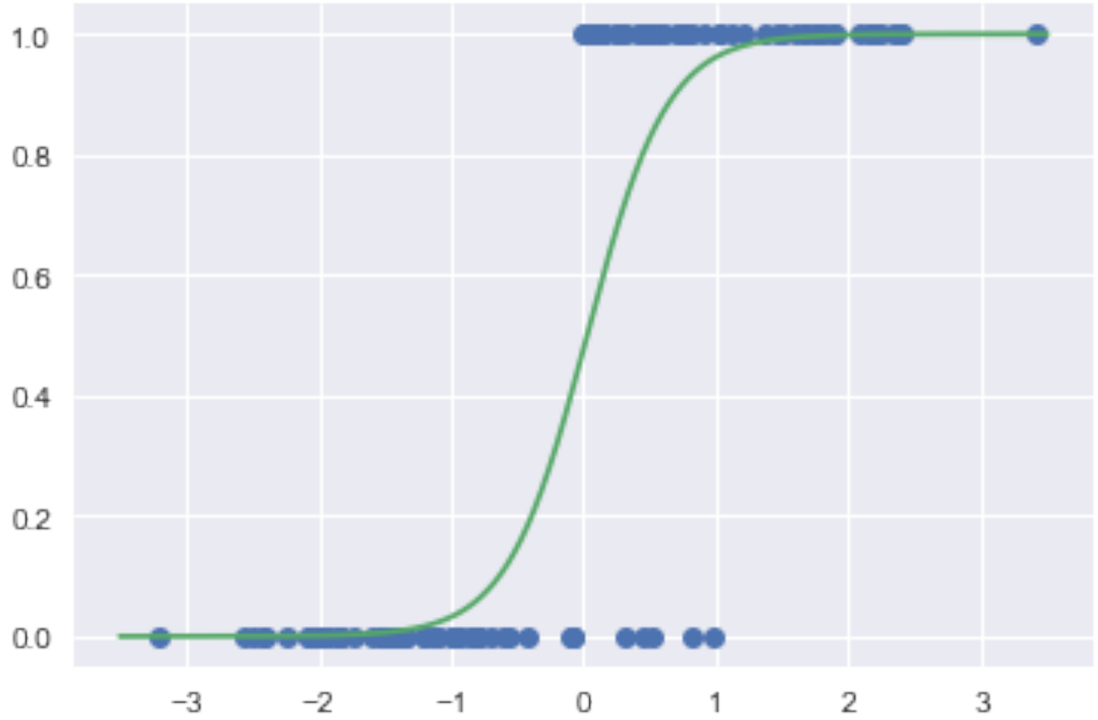
Hessian: $H = \mathbf{X}^T diag(p_i(1 - p_i))\mathbf{X}$ positive definite $\implies$ convex

Units based diagram



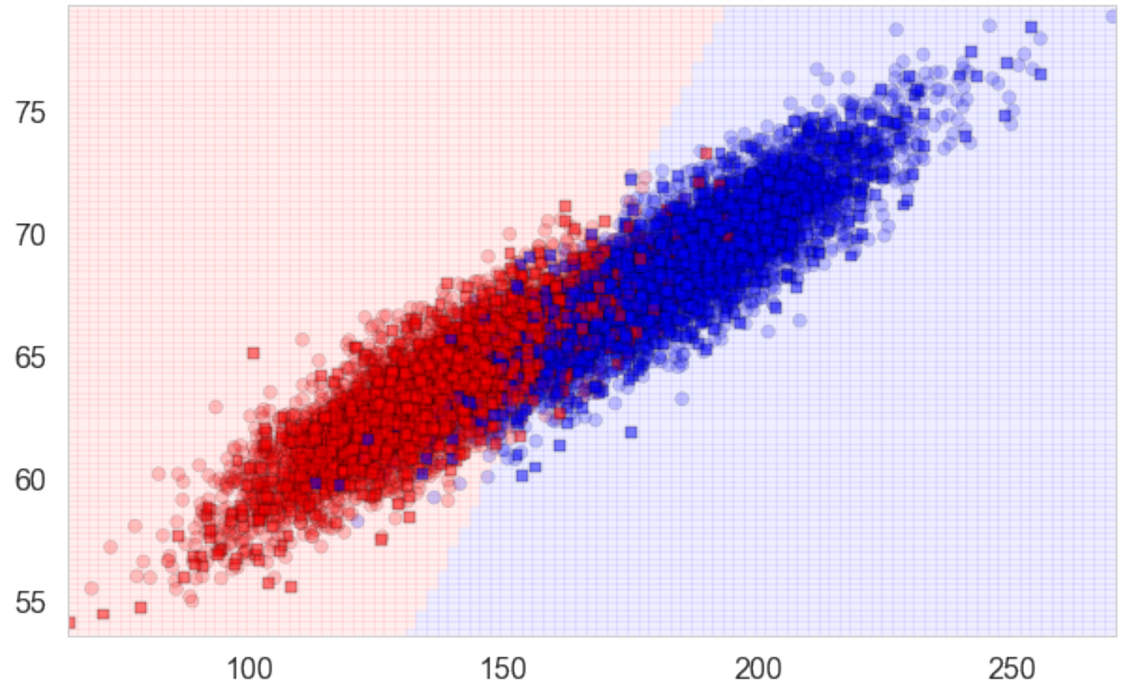
1-D Using Logistic regression

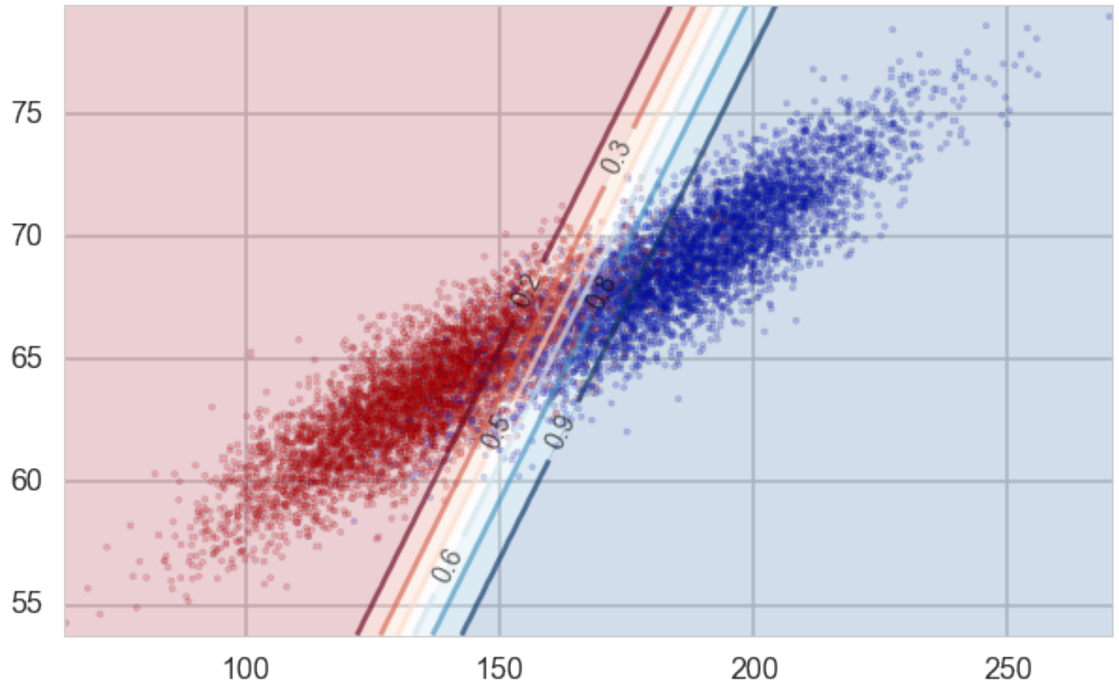




DISCRIMINATIVE CLASSIFIER

$P(y|x) : P(male|height, weight)$





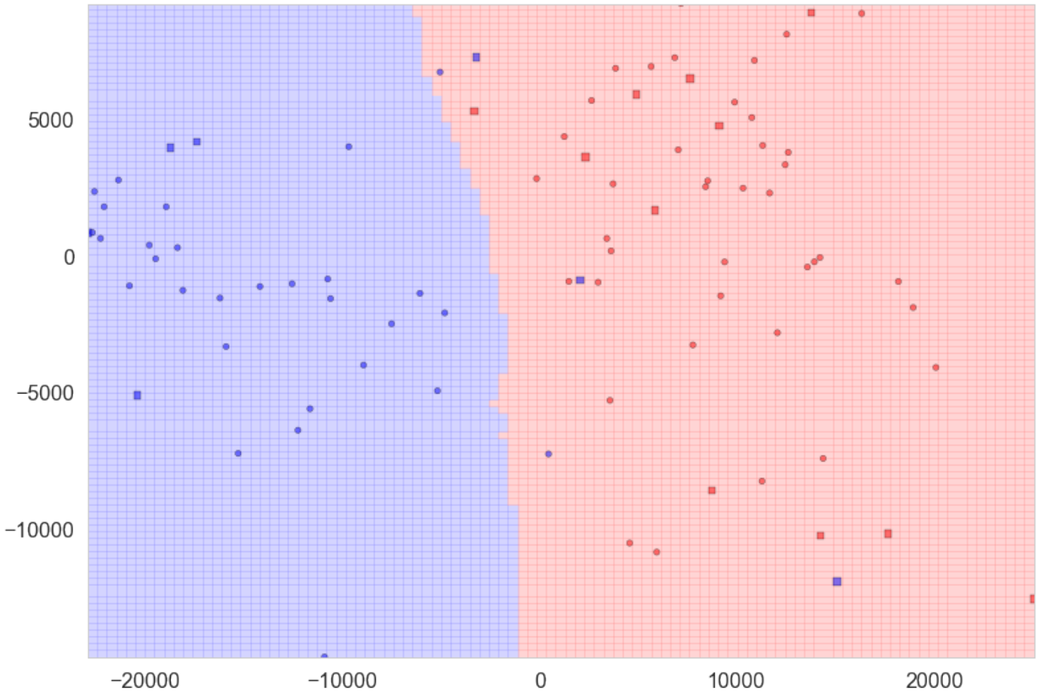
Discriminative Learning

- are these classifiers any good?
- they are discriminative and draw boundaries, but thats it
- they are cheaper to calculate but shed no insight
- would it not be better to have a classifier that captured the generative process

4. EVALUATING CLASSIFIERS

Confusion Matrix

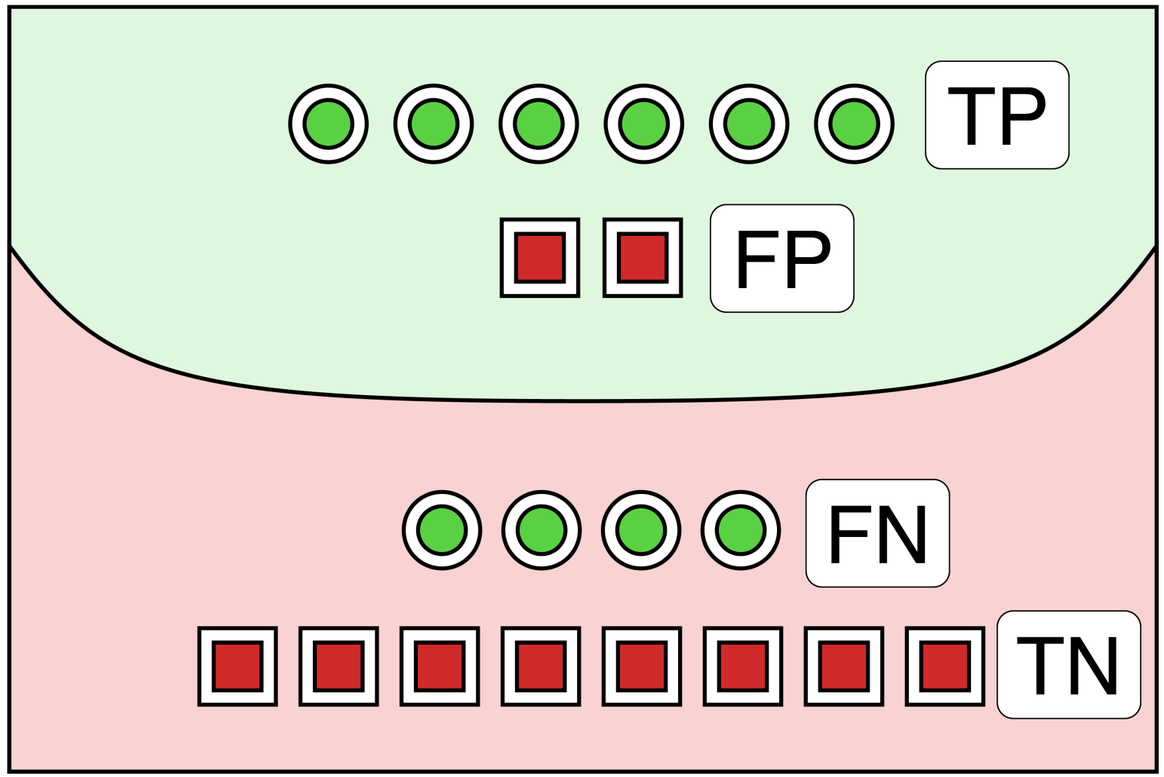
		Predicted		
		0	1	
Observed	0	TN True Negative	FP False Positive	ON Observed Negative
	1	FN False Negative	TP True Positive	OP Observed Positive
		PN Predicted Negative	PP Predicted Positive	



Metrics (from Glassner)

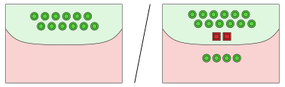
- accuracy is a number from 0 to 1. It's a general measure of how often the prediction is correct.
- visually looking at the confusion matrix is another important way to evaluate
- Precision (also called positive predictive value, or PPV) tells us the percentage of our samples that were properly labeled "positive," relative to all the samples we labeled as "positive." Numerically, it's the value of TP relative to TP+FP. In other words, precision tells us how many of the "positive" predictions were really positive.
- recall, (also called sensitivity, hit rate, or true positive rate). This tells us the percentage of the positive samples that we correctly labeled.
- F1 score is the harmonic mean of precision and recall. Generally speaking, the f1 score will be low when either precision or recall is low, and will approach 1 when both measures also approach 1.

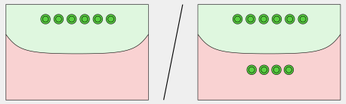
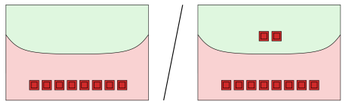
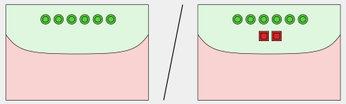
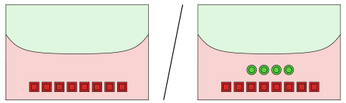
Metrics (example)



Lets understand the various metrics on this example

Simple metrics

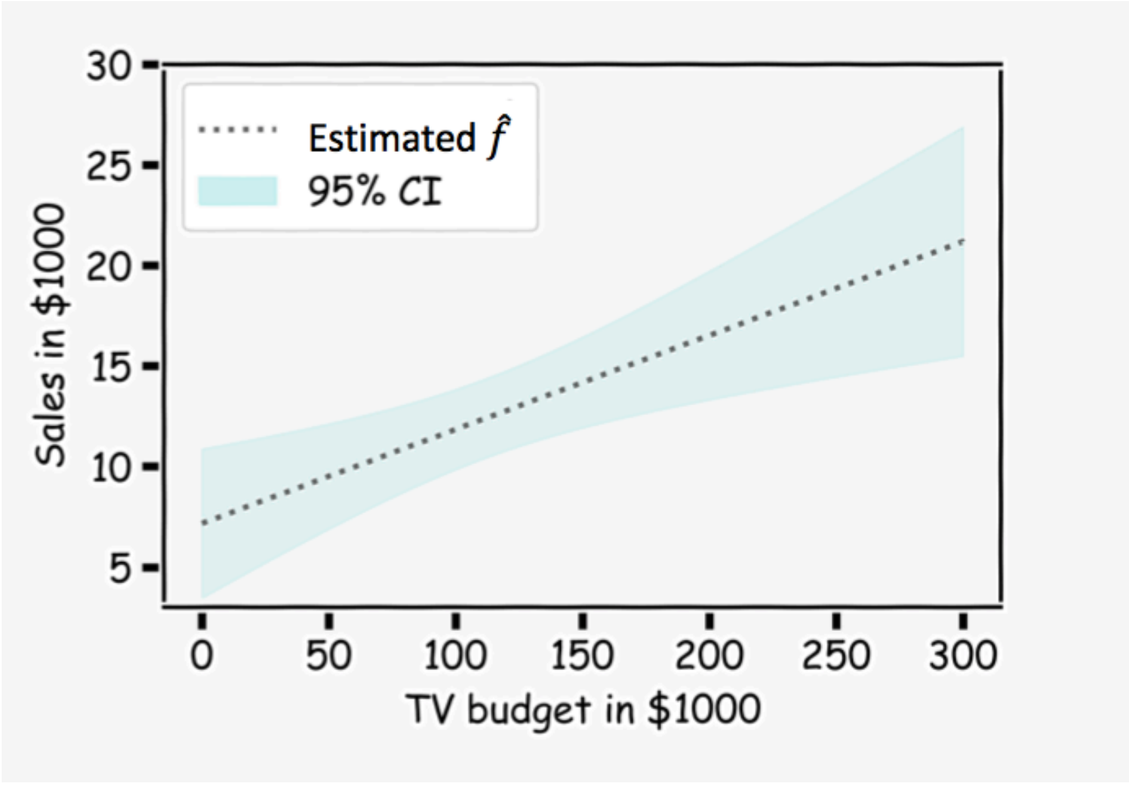
Accuracy	ACC	$\frac{TP+TN}{TP+FP+TN+FN}$		=	$\frac{6+8}{6+2+4+8}$	= 14/20	= 0.7	
F1 Score	F1	$\frac{2\,TP}{2\,TP+FP+FN}$		=	$\frac{2*6}{(2*6)+2+4}$	= 12/18	≈ 0.66	

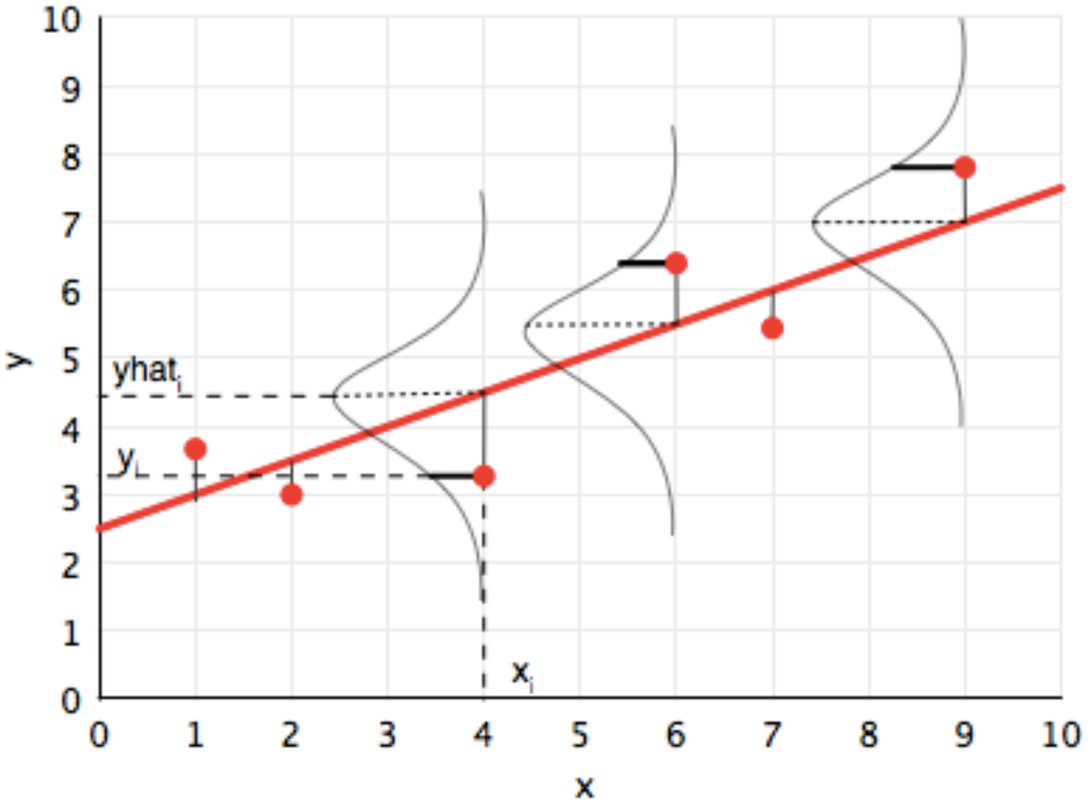
Recall	TPR	$\frac{TP}{TP+FN}$		=	$\frac{6}{6+4}$	= 6/10	= 0.6	
Specificity	TNR	$\frac{TN}{TN+FP}$		=	$\frac{8}{8+2}$	= 8/10	= 0.8	
Precision	PPV	$\frac{TP}{TP+FP}$		=	$\frac{6}{6+2}$	= 6/8	= 0.75	
Negative Predictive Value	NPV	$\frac{TN}{TN+FN}$		=	$\frac{8}{8+4}$	= 8/12	≈ 0.66	

5. Back to Regression: Prediction and the mean

Prediction vs Possible Prediction

- In machine learning we do not care too much about the functional form of our prediction $\hat{y} = \hat{f}(x)$, as long as we predict "well"
- Remember however our origin story for the data: the measured y is assumed to have been a draw from a gaussian distribution at each x: this means that our prediction at an as yet not measured x should also be a draw from such a gaussian
- Still, we use the mean value of the gaussian as the value of the "prediction", but note that we can have many "predicted" data sets, all consistent with the original data we have
- This means that there is an additional "smear" to that of the regression line...

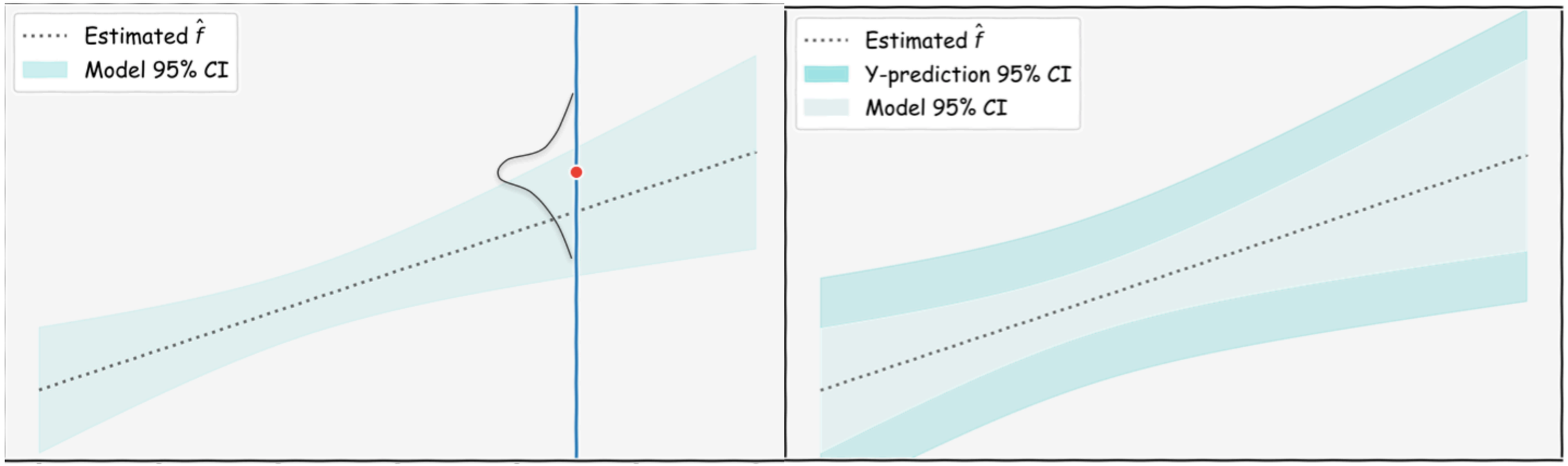

Estimated $\hat{f}$
95% CI
Sales in $1000
TV budget in $1000
30
25
20
15
10
5
0
50
100
150
200
250
300


yhat$_i$
y$_i$
x$_i$
y
x
0
1
2
3
4
5
6
7
8
9
10

From Likelihood to Predictive Distribution

- the band on the previous graph is the sampling distribution of the regression line, or a representation of the sampling distribution of the $\mathbf{w}$.
- $p(y|\mathbf{x}, \mu_{MLE}, \sigma^2_{MLE})$ is a probability distribution
- thought of as $p(y^*|\mathbf{x}^*, \{\mathbf{x}_i, y_i\}, \mu_{MLE}, \sigma^2_{MLE})$, it is a predictive distribution for as yet unseen data y^* at $\mathbf{x}^*$, or the sampling distribution for data, or the data-generating distribution, at the new covariates $\mathbf{x}^*$. This is a wider band.

Mean vs Prediction



The two risks, or rather risk and score

When we estimate a model using maximum likelihood converted to a risk (how? by NLL) we are calling this risk an **estimation risk**.

Scoring is a different enterprise, where we want to compare different models using their **score** or **decision risk**

The latter leads to the idea of the **Bayes Model**, the best you can do..

The notion of Bayes Risk

$$R^* = \inf_h R_{out}(h) = \inf_h \int dx p(x,y)(h(x)-y)^2.$$

Its the minimum risk **ANY** model can achieve.

Want to get as close to it as possible.

Could infimum amongst all possible functions. OVERFITTING!

Instead restrict to a particular Hypothesis Set: $\mathcal{H}$.

Bayes Risk for Regression

$$R_{out}(h) = \int dx p(x,y)(h(x)-y)^2.$$

$$= E_X E_{Y|X}[(h-y)^2] = E_X E_{Y|X}[(h-r+r-y)^2]$$

where $r(x) = E_{Y|X}[y]$ is the "regression" function.

$$R_{out}(h) = E_X[(h-r)^2] + R^*; R^* = E_X E_{Y|X}[(r-y)^2]$$

For 0 mean, finite variance, then, σ^2, the noise of ϵ, is the Bayes Risk, also called the irreducible error.

Note that:

- We are never given a population, rather we get a training set.
- Now, varying training sets make $R_{out}(h)$ a stochastic quantity, varying from one training set to another, since a different model is fit on each set!
- **Goal of Learning**: Build a function whose risk is closest to Bayes Risk, on our training set

$$\langle R \rangle = E_{\mathcal{D}}[R_{out}(g_{\mathcal{D}})] = E_{\mathcal{D}} E_{p(x,y)}[(g_{\mathcal{D}}(x) - y)^2]$$

$\bar{g} = E_{\mathcal{D}}[g_{\mathcal{D}}] = (1/M) \sum_{\mathcal{D}} g_{\mathcal{D}}$. Then,

$$\langle R \rangle = E_{p(x)}[E_{\mathcal{D}}[(g_{\mathcal{D}} - \bar{g})^2]] + E_{p(x)}[(f - \bar{g})^2] + \sigma^2$$

where $y = f(x) + \epsilon$ is the true generating process and ϵ has 0 mean and finite variance σ^2.

$$\langle R \rangle = E_{p(x)}[E_{\mathcal{D}}[(g_{\mathcal{D}} - \bar{g})^2]] + E_{p(x)}[(f - \bar{g})^2] + \sigma^2$$

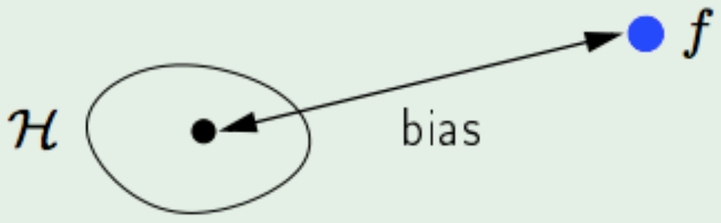
This is the bias variance decomposition for regression.

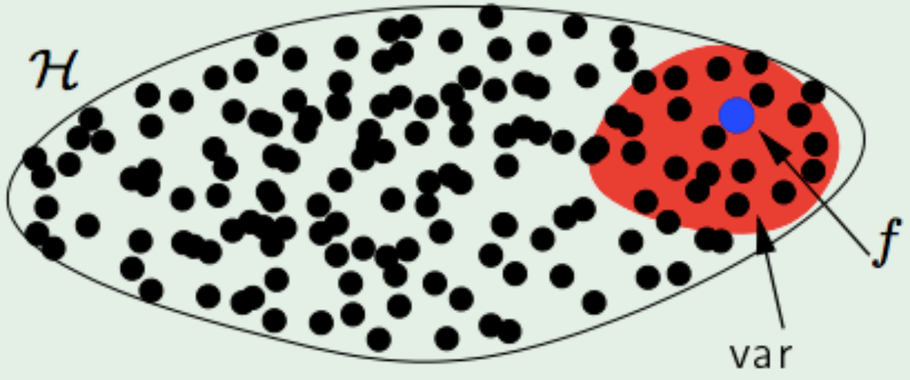
- first term is **variance**, squared error of the various fit g's from the average g, the hairiness.
- second term is **bias**, how far the average g is from the original f this data came from.
- third term is the **stochastic noise**, minimum error that this model will always have.

The tradeoff

$$\text{bias} = \mathbb{E}_{\mathbf{x}}\left[\left(\bar{g}(\mathbf{x}) - f(\mathbf{x})\right)^2\right]$$

$$\text{var} = \mathbb{E}_{\mathbf{x}}\left[\mathbb{E}_{\mathcal{D}}\left[\left(g^{(\mathcal{D})}(\mathbf{x}) - \bar{g}(\mathbf{x})\right)^2\right]\right]$$





$\downarrow$ $\mathcal{H}\uparrow$ $\uparrow$

To recap: Empirical Risk Minimization

- We dont know $p(x, y)$, otherwise why are we bothering? (we dont need to know it for the derivation)
- And what we do is Fit a hypothesis $h = g_{\mathcal{D}}$, where $\mathcal{D}$ is our training sample.
- We use the empirical distribution on our sample as our best estimate of $p(x, y)$
- $\hat{p}(x, y) = \frac{1}{N} \sum_{i \in \mathcal{D}} \delta(x - x_i)\delta(y - y_i)$
- Then $R_{out}(g_{\mathcal{D}}) = \frac{1}{N} \sum_{i \in \mathcal{D}} (g_{\mathcal{D}i}(x_i) - y_i)^2$ and minimize to get best for $g_{\mathcal{D}}$

Classification Risk

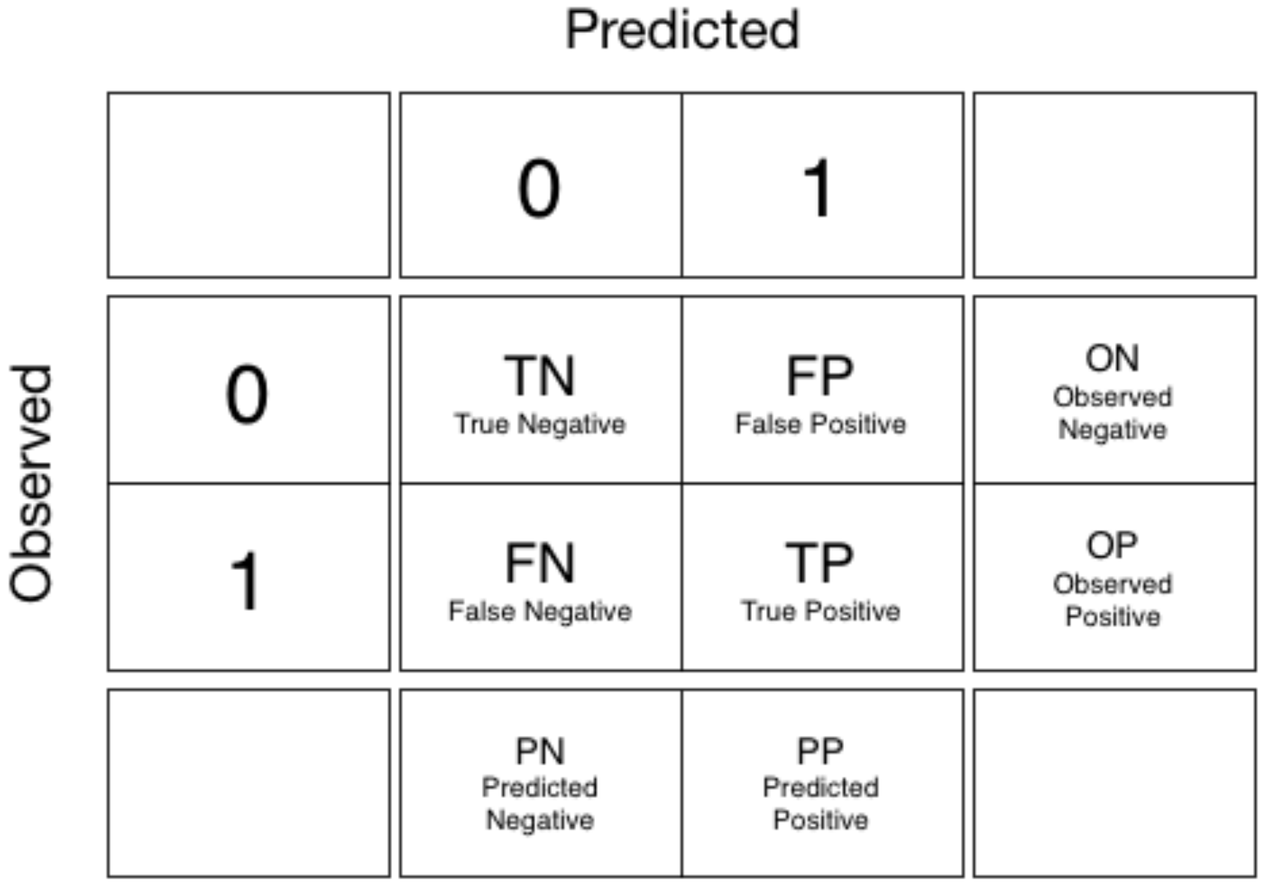
$$R_a(x) = \sum_y l(y, a(x))p(y|x)$$

That is, we calculate the **predictive averaged risk** over all choices y, of making choice a for a given data point.

Overall risk, given all the data points in our set:

$$R(a) = \sum_x R_a(x)$$

Two class Classification



$$R_a(x) = l(1, g)p(1|x) + l(0, g)p(0|x).$$

Then for the "decision" $a = 1$ we have:

$$R_1(x) = l(1, 1)p(1|x) + l(0, 1)p(0|x),$$

and for the "decision" $a = 0$ we have:

$$R_0(x) = l(1, 0)p(1|x) + l(0, 0)p(0|x).$$

CLASSIFICATION RISK

- $R_{g,\mathcal{D}}(x) = P(y_1|x)\ell(g, y_1) + P(y_0|x)\ell(g, y_0)$
- The usual loss is the 1-0 loss $\ell = 1_{g \neq y}$. (over all points $\frac{1}{n}\sum_{i}^{n} I\left(y_i = \hat{y}_i\right)$)
- Thus, $R_{g=y_1}(x) = P(y_0|x)$ and $R_{g=y_0}(x) = P(y_1|x)$

CHOOSE CLASS WITH LOWEST RISK

1 if $R_1 \leq R_0 \implies$ 1 if $P(0|x) \leq P(1|x)$.

choose 1 if $P(1|x) \geq 0.5$! Intuitive!