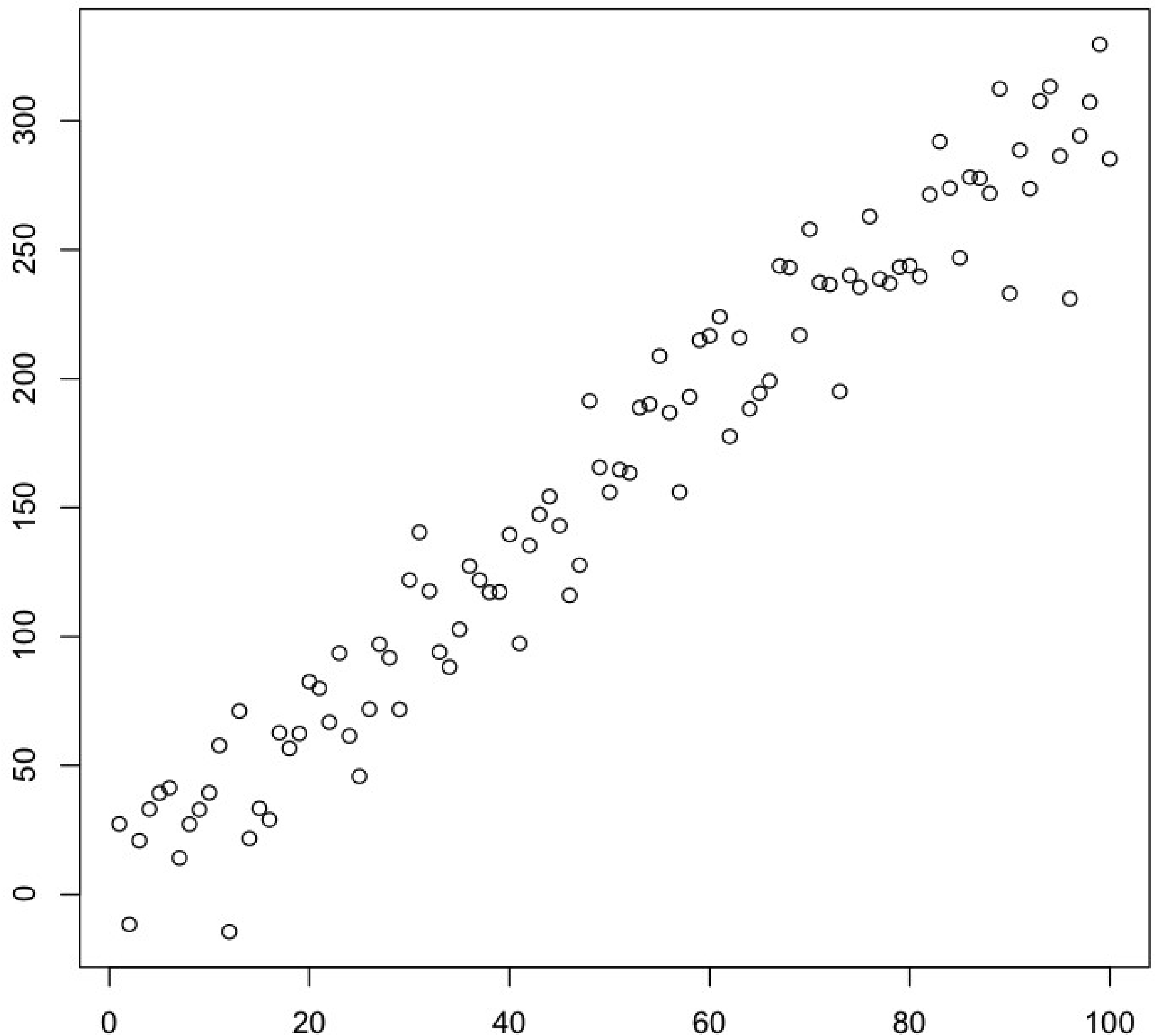
Distributed GLM Implementation

on H2O platform

Tomas Nykodym, 0xDATA

Linear Regression



Data:

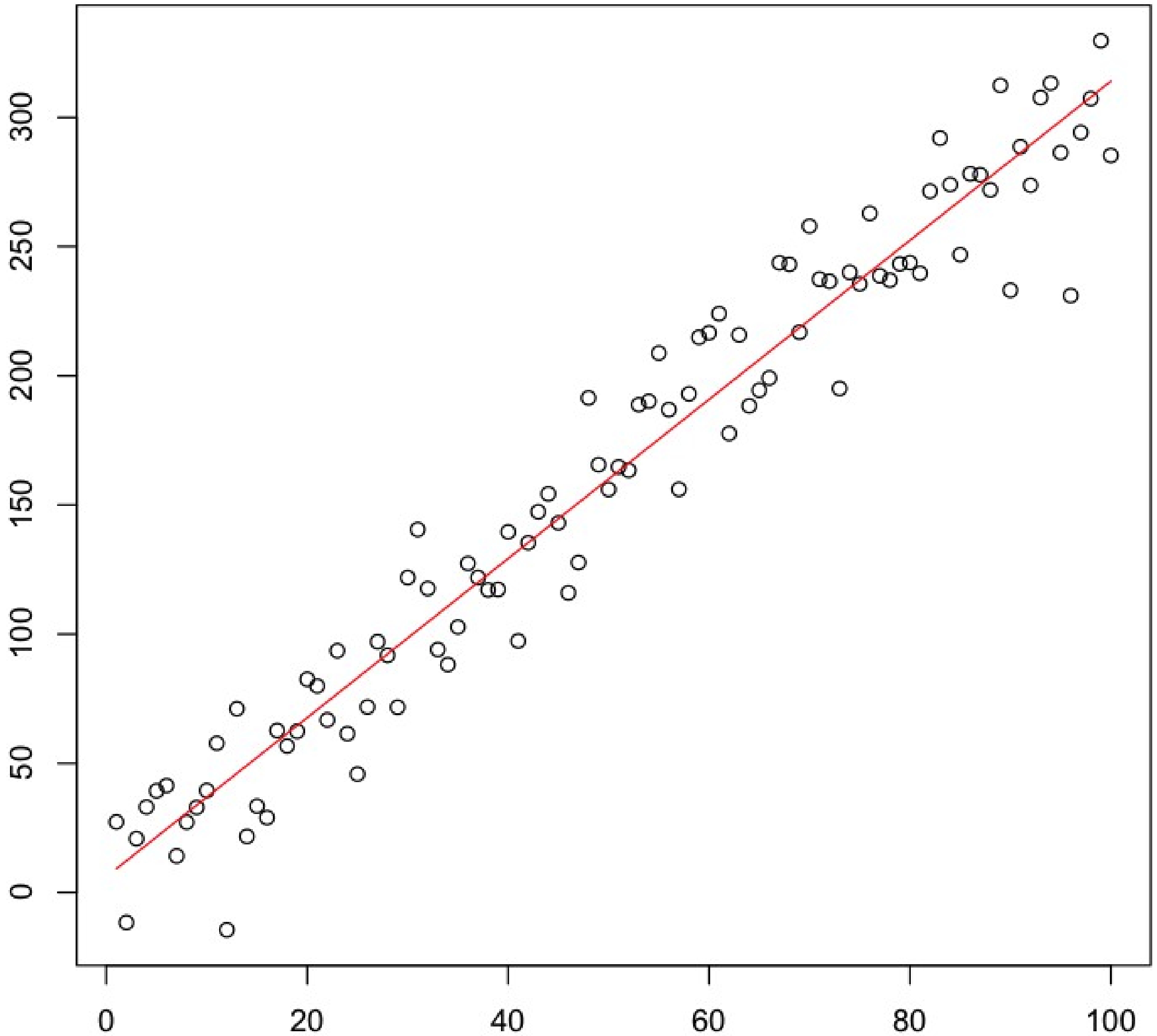
x, y + noise

Goal:

predict y using x

i.e. find a,b s.t.
y = a*x + b

Linear Regression
Least Squares Fit



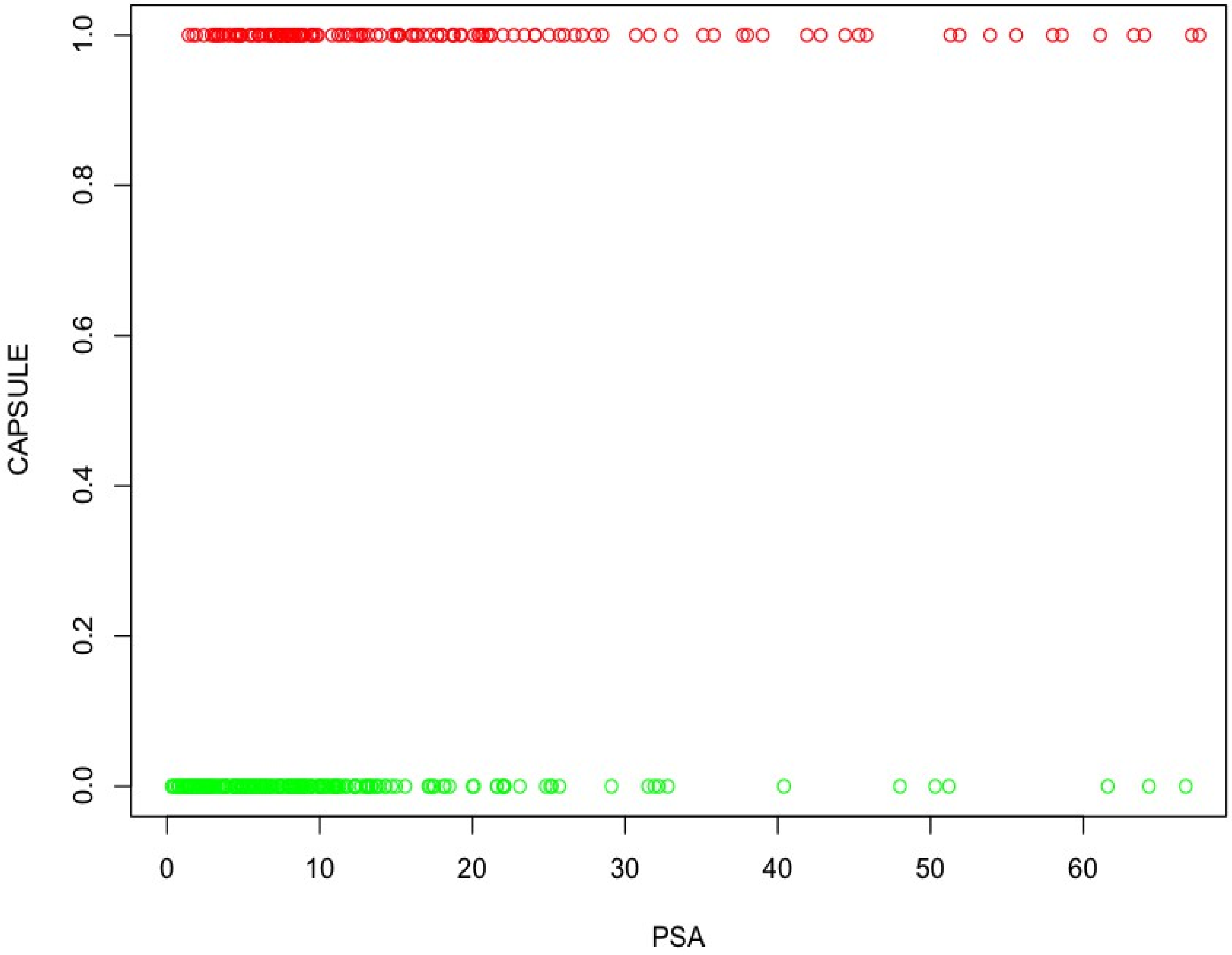
Real Relation:

$y=3x+10+N(0,20)$

Best Fit:

$y = 3.08*x + 6$

Prostate Cancer Example



Data:

x = PSA

(prostate-specific antigen)

y = CAPSULE

0 = no tumour

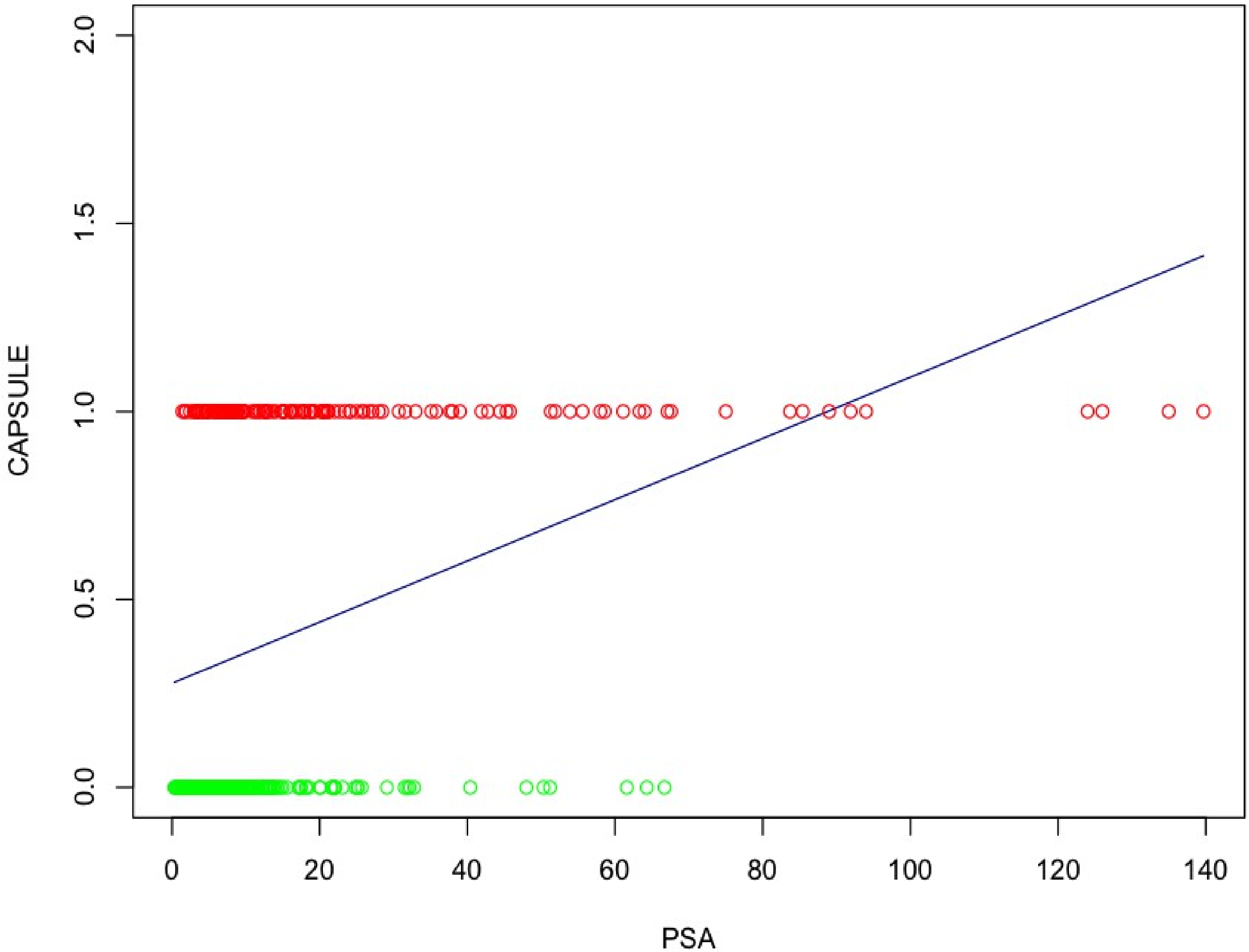
1 = tumour

Goal:

predict y using x

Prostate Cancer Example

Linear Regression Fit



Data:

x = PSA

(prostate-specific antigen)

y = CAPSULE

0 = no tumour

1 = tumour

Fit:

Least squares fit

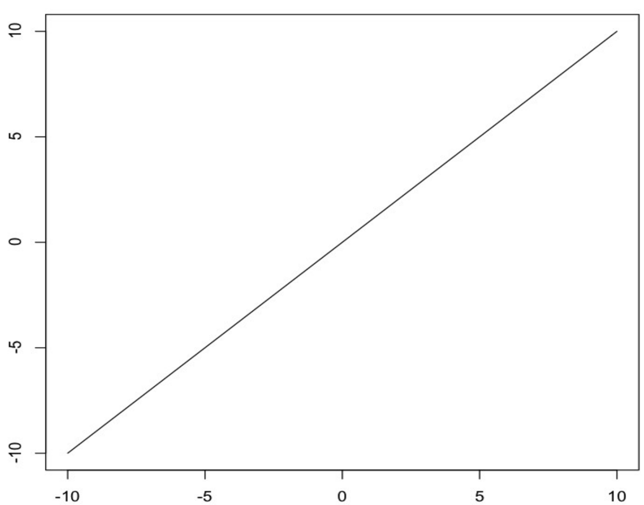
Generalized Linear Model

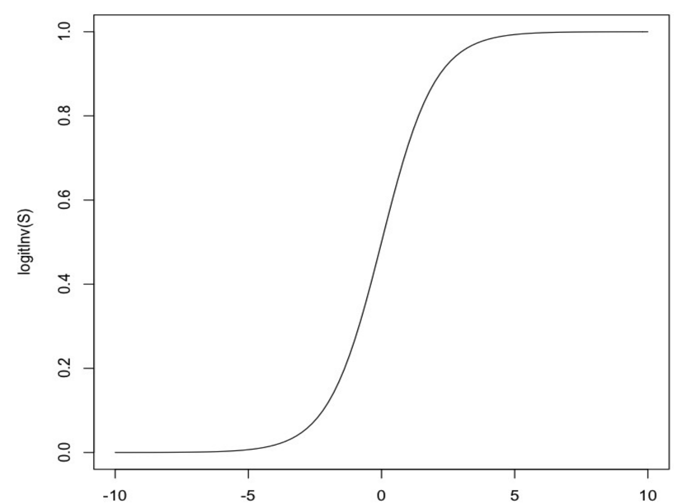
Generalizes linear regression by:

- adding a link function g to transform the output

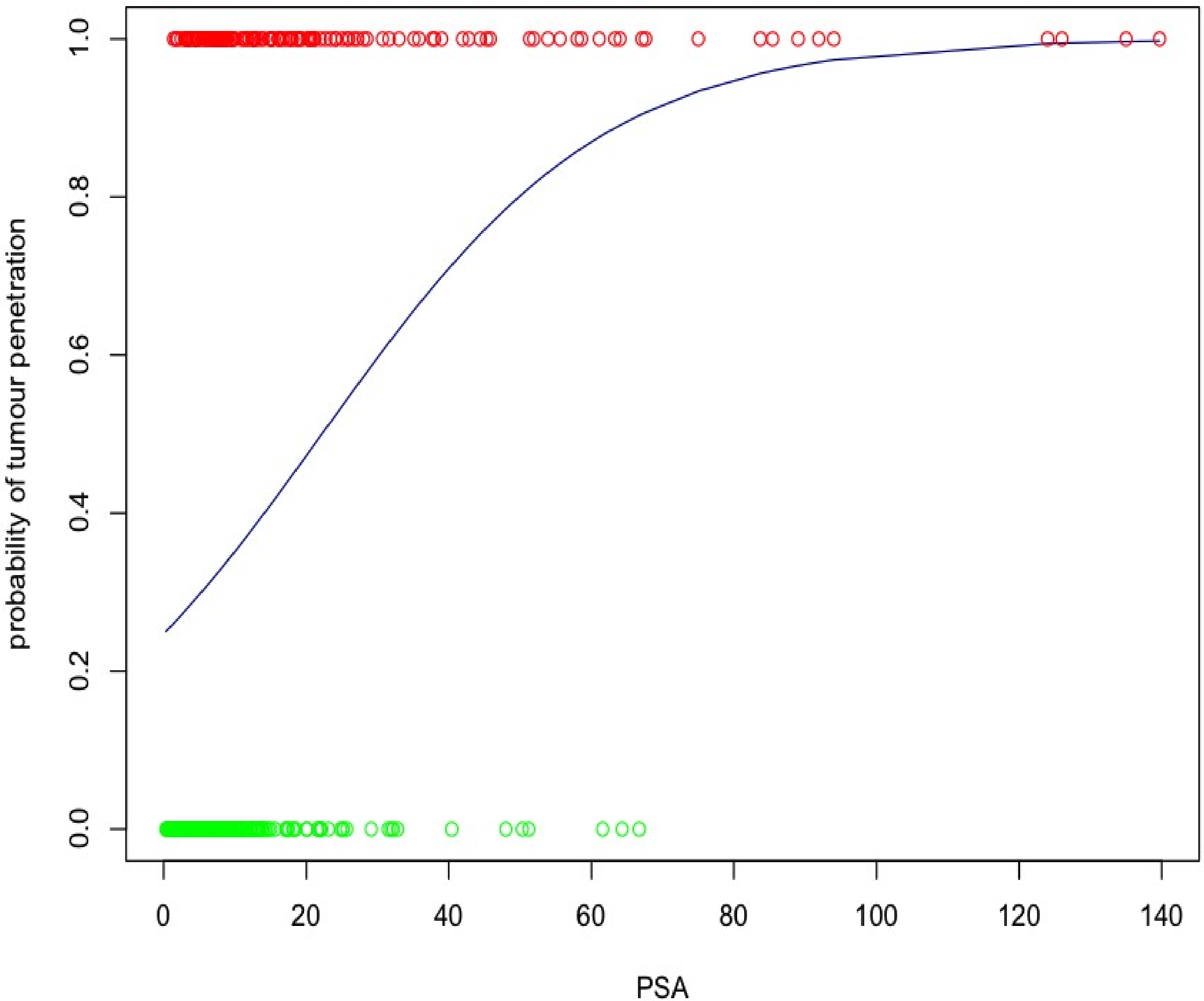
  z = g(y) – new response variable

- noise (i.e.variance) does not have to be constant
- fit is maximal likelihood instead of least squares





Prostate Cancer
Logistic Regression Fit



Data:

x = PSA

(prostate-specific antigen)

y = CAPSULE

0 = no tumour

1 = tumour

GLM Fit:

- **Binomial** family
- **Logit** link
- Predict probability of CAPSULE=1.

Implementation - Solve GLM by IRLSM

Input:

- X: data matrix N*P
- Y: response vector (N rows)
- family, link function, α,β

Output:

- β vector of coefficients, solution to max-likellihood

OUTER LOOP:

While β changes, compute:

$$z_{k+1} = \beta_k + (y - \mu_k)\frac{d\eta}{d\mu}$$

$$W_{k+1}^{-1} = \left(\frac{d\eta}{d\mu}\right)^2 Var(\mu_k)$$

$$XX = X^T W_{k+1} X$$

$$Xz = X^T W z_{k+1}$$

INNER LOOP:

Solve elastic net:

ADMM(Boyd 2010, page 43):

$$\gamma^{l+1} = (X^T W X + \rho I)^{-1} X^T W z + \rho(\beta^l - u^l)$$

$$\beta^{l+1} = S_{\lambda/\rho}(\gamma^{l+1} + u^l)$$

$$u^{l+1} = u^k + \gamma^{l+1} - \beta^{l+1}$$

H2O Implementation

Outer Loop:

(Map Reduce Task)

```
public class SimpleGLM extends MRTask {

  @Override public void map(Chunk c) {
    res = new double [p][p];
    for(double [] x:c.rows()){
      double eta,mu,var;
      eta = computeEta(x);
      mu = _link.linkInv(eta);
      var = Math.max(1e-5,_family.variance(mu));
      double dp = _link.linkInvDeriv(eta);
      double w = dp*dp/var;
      for(int i = 0; i < x.length; ++i)
        for(int j = 0; j < x.length; ++j)
          res[i][j] += x[i]*x[j]*w;
    }
  }
  @Override public void reduce(SimpleGLM g) {
    for(int i = 0; i < res.length; ++i)
      for(int j = 0; i < res.length; ++i)
        res[i][j] += g.res[i][j];
  }
}
```

Inner Loop:

(ADMM solver)

```
public double [] solve(Matrix xx, Matrix xy) {
  // ADMM LSM Solve
  CholeskyDecomposition lu; // cache decomp!
  lu = new CholeskyDecomposition(xx);
  for( int i = 0; i < 1000; ++i ) {
    // Solve using cached Cholesky decomposition!
    xm = lu.solve(xyPrime);
    // compute u and z update
    for( int j = 0; j < N-1; ++j ) {
      double x_hat = xm.get(j, 0);
      x_norm += x_hat * x_hat;
      double zold = z[j];
      z[j] = shrinkage(x_hat + u[j], kappa);
      u[j] += x_hat - z[j];
      u_norm += u[j] * u[j];
    }
  }
}

double shrinkage(double x, double  k) {
  return Math.max(0,x-k)-Math.max(0,-x-k);
}
```

Regularization

Elastic Net (Zhou, Hastie, 2005):

$$\beta = argmin\left(X\beta - y\right)^T\left(X\beta - y\right) + \alpha\,\|\beta\|_1 + \left(1-\alpha\right)\|\beta\|_2^2$$

- Added L1 and L2 penalty to β to:
  - avoid overfitting, reduce variance
  - obtain sparse solution (L1 penalty)
  - avoid problems with correlated covariates

No longer analytical solution.

Options: LARS, ADMM, Generalized Gradient, ...

Linear Regression
Least Squares Method

Find β by minimizing the sum of squared errors:

$$\beta = argmin(X\beta - y)^T(X\beta - y)$$

Analytical solution:

$$\beta = (X^T X)^{-1} X^T y = \left(\frac{1}{n}\sum x_i x_i^T\right)^{-1} \frac{1}{n}\sum x_i y$$

Easily parallelized if X^TX is reasonably small.

Generalized Linear Model

- Generalizes linear regression by:
  - adding a link function g to transform the response

    $z = g(y)$ – new response variable

    $\eta = X\beta$ – linear predictor

    $\mu = g^{-1}(\eta)$
  - y has a distribution in the exponential family
  - variance depends on μ

    e.g $var(\mu) = \mu*(1-\mu)$ for Binomial family.
  - fit by maximizing the likelihood of the model