EC999: Classification

Thiemo Fetzer

University of Chicago & University of Warwick

April 27, 2017

Regression versus Classification

- Classification refers to cases where the y_i is a categorical variable, such as Eye color, Gender, Brand, Sentiment = Positive, Negative, Neutral.
- Regression refers to cases where the dependent variable is numeric, like price, quantity, ...
- There are cases, where categorical variables can be given a numeric interpretation, e.g. a categorical variable with two levels can be expressed as a dummy variable/ binary variable, such as Gender = 1 if male, Gender = 0 if female.

Classification Example (1): Land Use Patterns

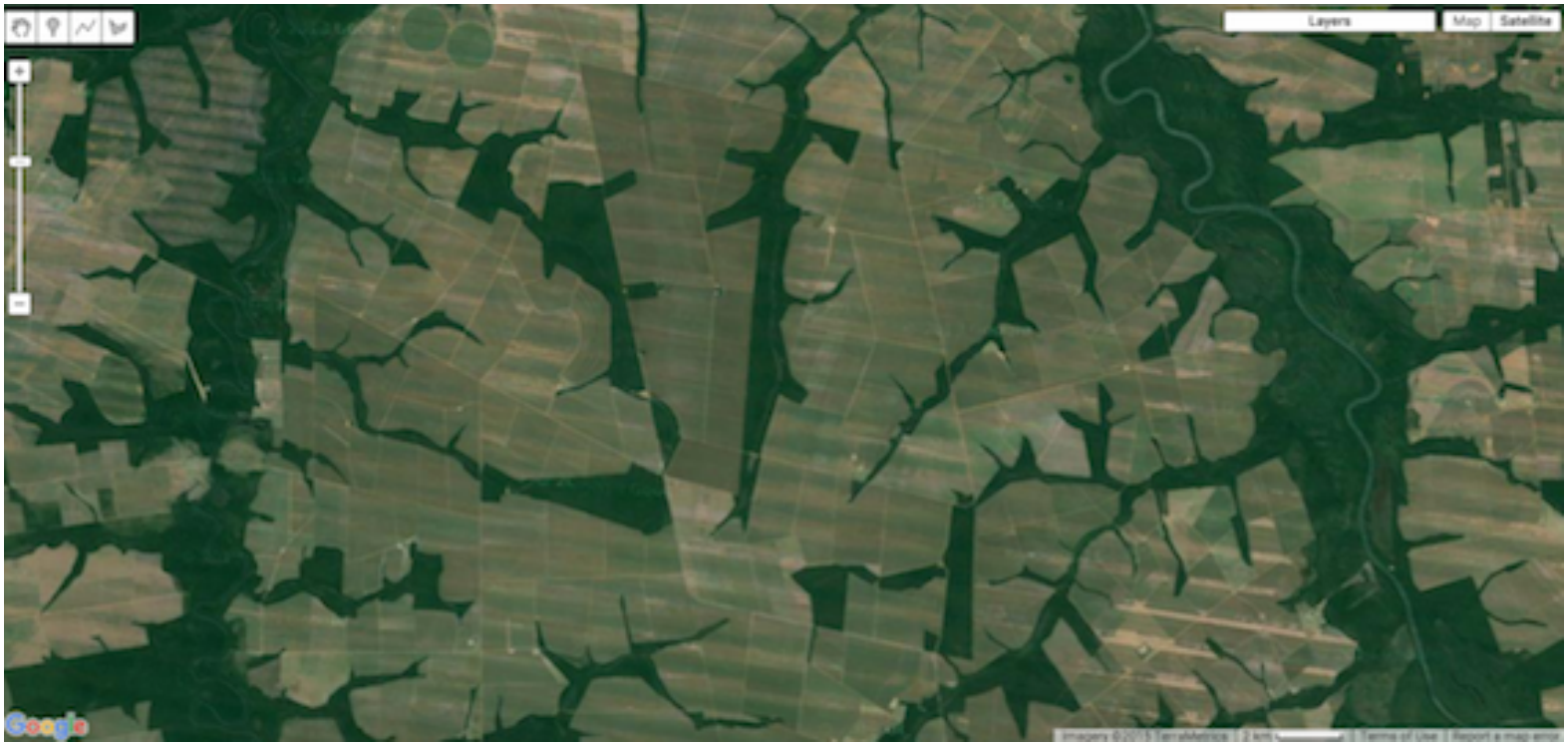


Figure: Classifiying pixels from satellite imagery into common land use clusters: Cropland, Forest and Shrubland as used in Fetzer and Marden (2016).

Classification Example (1): Land Use Patterns

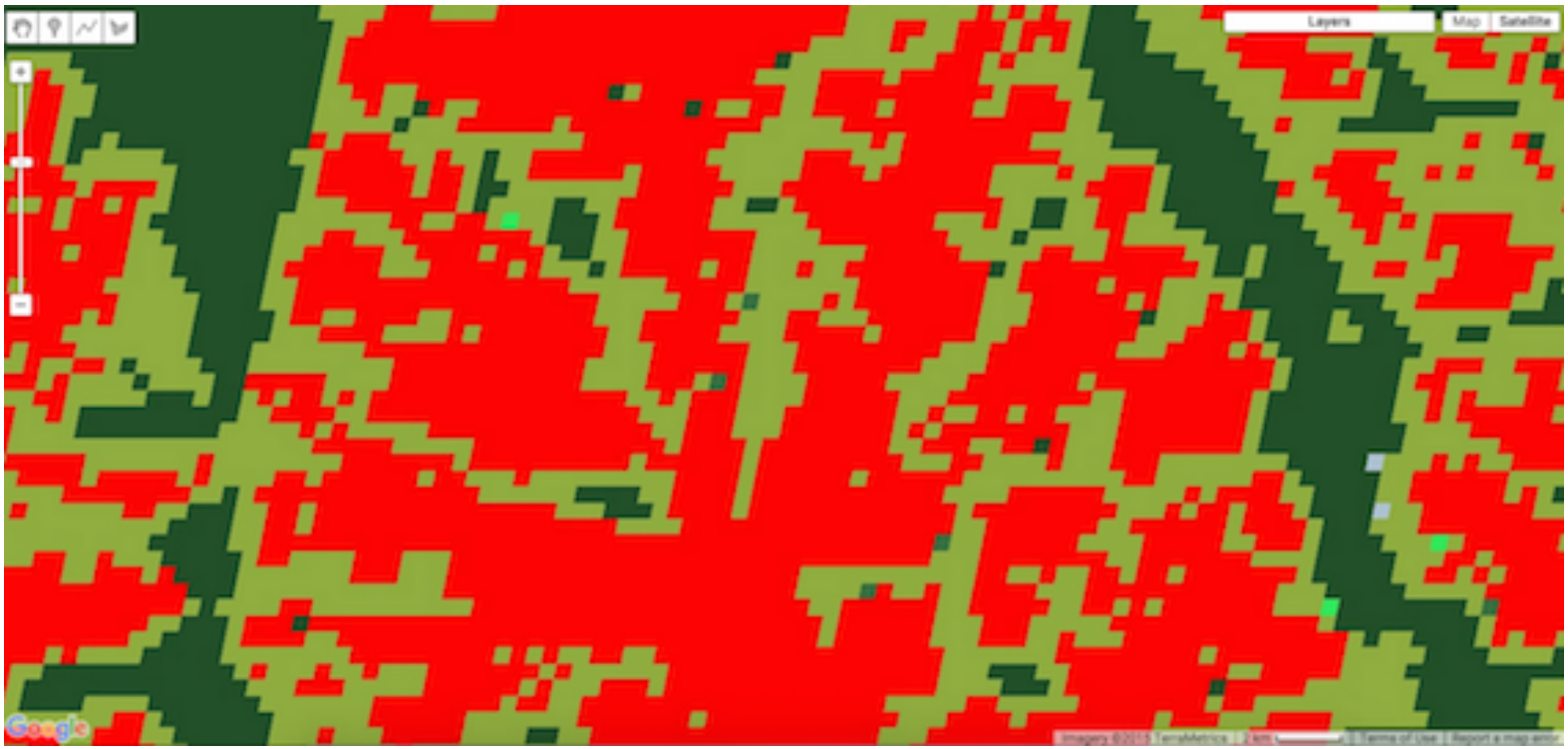


Figure: Classifiying pixels from satellite imagery into common land use clusters: Cropland, Forest and Shrubland as used in Fetzer and Marden (2016).

Other Classification Examples

1. Fraud detection in financial transaction data
2. News Feed categorization
3. Email Spam detection
4. Facebook Image Upload Nudity detection
5. Sentiment categorization

The General Classification Problem

- A qualitative variable takes a value in set $\mathcal{C}$.
- We will denote $|\mathcal{C}| = c$ the number of different categories.
- We observe a feature vector X and a responsive variable Y that takes on values from the set $\mathcal{C}$
- We estimate for each possible value $y \in \mathcal{C}$:

$$\hat{P}(Y = y|X)$$

- Following a decision rule, we will assign the "correct" label to that category or class y, which reduces some form of prediction error.
- The prediction error we can make here is the share of "wrongly" classified categories...
- How to measure "wrongly classified" or "correct" label?

Validation Set Approach to Assess Model Accuracy

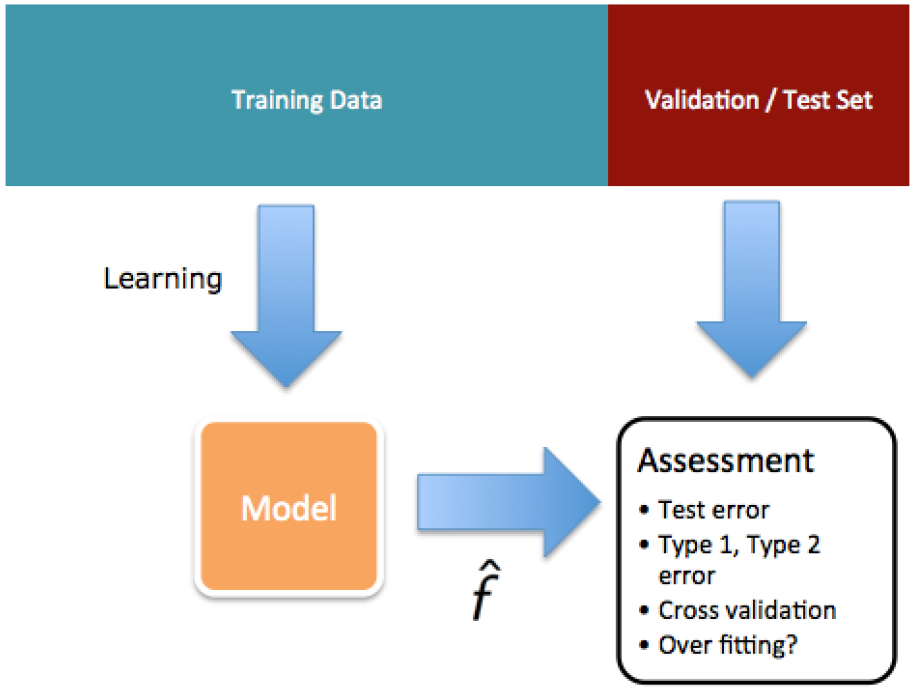


Figure: Validation Set Workflow Illustration.

Plan

Bayes Rule Reminder

Bayesian Decision Theory

Classification Error

Logistic Regression

Working with Text features

Bayes Theorem

- Bayes Rule States

$$\underbrace{P(Y|X)}_{posterior} = \frac{\overbrace{P(X|Y)}^{likelihood} \times \overbrace{P(Y)}^{prior}}{\underbrace{P(X)}_{marginal\ likelihood}}$$

- In case Y and X where independent, $P(Y|X) = P(Y)$, and $P(Y \cap X) = P(Y)P(X)$.
- X is the evidence, that tells you something about the probability of observing the data Y.

Bayes Theorem: Classical Problem

A patient takes a lab test and the result comes back positive. The test returns a correct positive result in only 98% of the cases in which the disease is actually present, and a correct negative result in only 97% of the cases in which the disease is not present. You know that 0.008 of the entire population have this disease.

Bayes Theorem: Classical Problem

What are we given? Let D be an indicator if the disaese is present and $+$ indicate whether the test is positive.

$$\begin{aligned} P(D) &= 0.008 \\ P(\neg D) &= 1 - 0.008 \\ P(+|D) &= 0.98 \\ P(-|D) &= 0.02 \\ P(+|\neg D) &= 0.03 \\ P(-|\neg D) &= 0.97 \end{aligned}$$

What is $P(D|+)$?

$$P(D|+) = \frac{P(D)P(+|C)}{P(+)}$$

Law of Total Probability

$$P(D|+) = \frac{P(D)P(+|D)}{P(+)}$$

Using the law of total probability

$$P(+) = P(+|D)P(D) + P(+|\neg D)P(\neg D)$$

$$= 0.98 \times 0.008 + 0.03 \times 0.992 = 0.0376$$

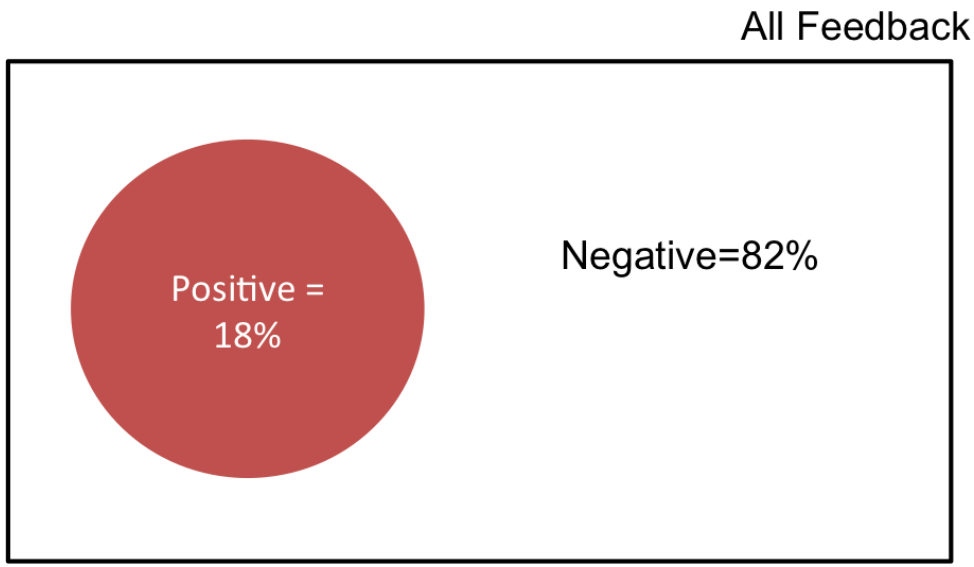
From this you get:

$$P(D|+) = \frac{P(D)P(+|D)}{P(+)}P(+) = \frac{0.008 \times 0.98}{0.0376} = 0.21$$

Plan

Prior Information

- Suppose you want to categorize some *teaching feedback* into two categories.
- In our example we have $Y \in \{\text{Positive}, \text{Negative}\}$.
- Suppose you know what the *priors* are, i.e. you know what the share of positive and negative feedbacks are *in the population*.



Best Decision Without Additional Information

- Suppose you now receive a new feedback $\tilde{Y}$, and you want to categorize it *without any additional information*
- What decision / assignment rule would minimize the missclassification error?

Clearly this should be

$$\tilde{Y} = \begin{cases} + & \text{if } P(+) > P(-) \\ - & \text{if } P(+) < P(-) \end{cases}$$

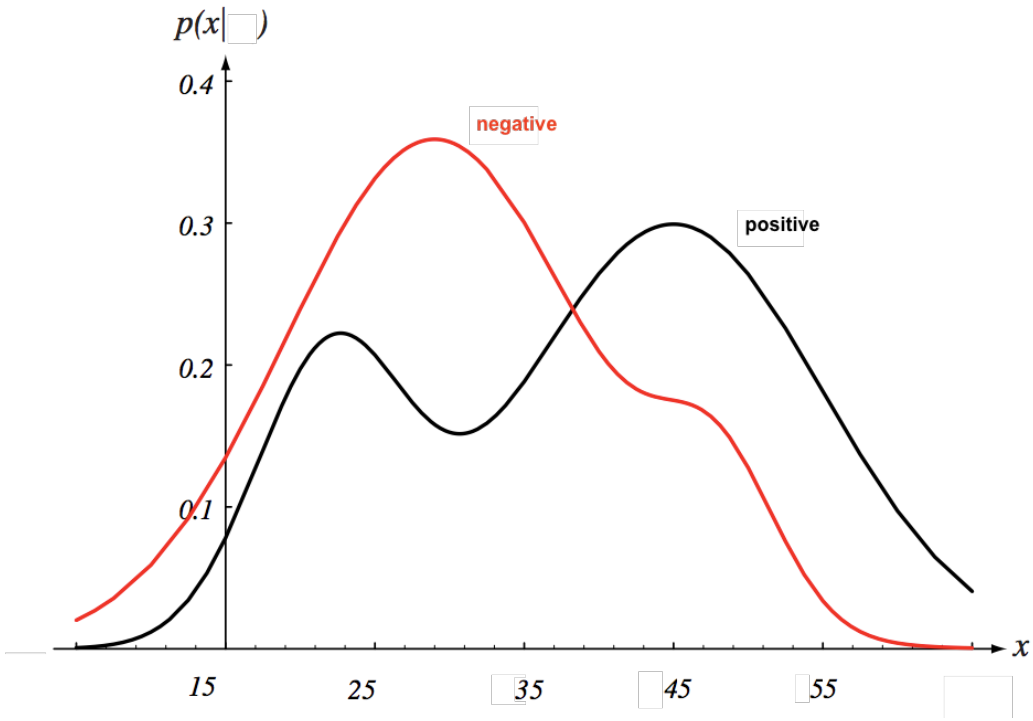
What is the error that we make with this *decision rule*?

$$P(error) = \min\{P(+), P(-)\}$$

So here, we would assign any feedback coming in as negative $-$; the maximum error we can make is 18%.

Best Decision With Additional Information

- Suppose you now have additional information X, let that information be the number of characters contained in a feedback.
- Suppose that the distribution of the number of characters is smooth in the population.
- Plot the density function $p(x|+)$ and $p(x|-)$.



Using Bayes Rule

- For every value of x, we can compute the *posterior* probability using Bayes rule.

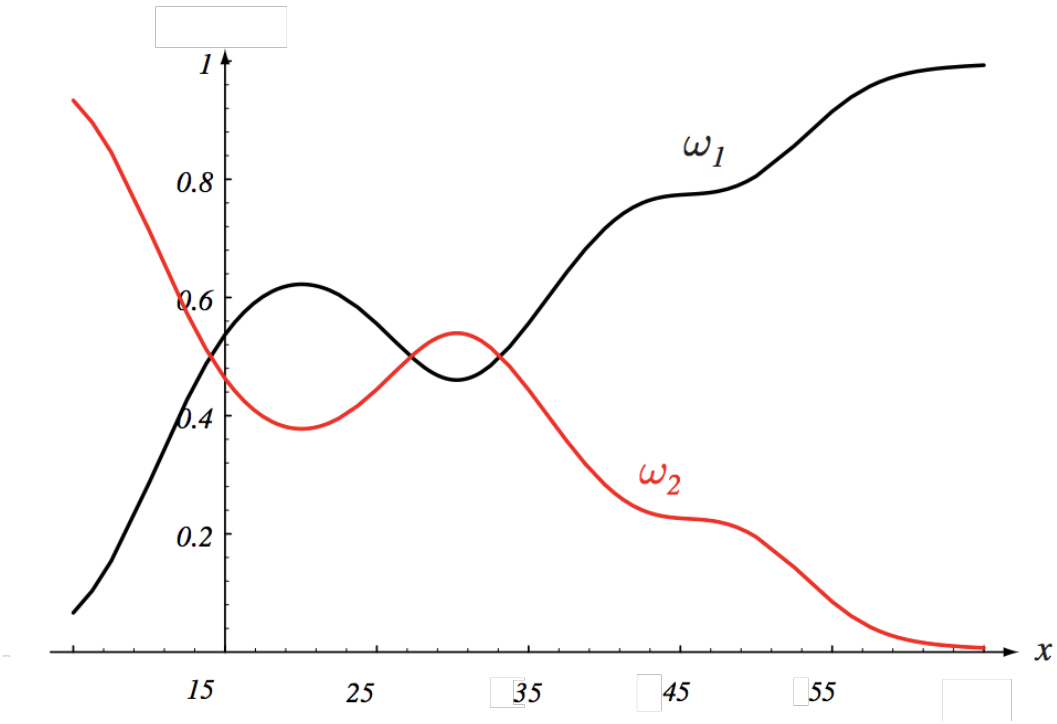
$$\underbrace{P(+|x)}_{posterior} = \frac{\overbrace{p(x|+)}^{likelihood} \times \overbrace{P(+)}^{prior}}{\underbrace{p(x)}_{marginal\ likelihood}}$$

where

$$p(x) = p(x|+)P(+) + p(x|-)P(-)$$

Using Bayes Rule

- Plot the posterior densities $p(+|x)$ and $p(-|x)$.



- How would you now decide on which label to assign? I.e. what is the decision rule that minimzes test error?

MAP Decision Rule

- It seems intuitive, that you should assign

$$\tilde{Y} = \begin{cases} + & \text{if } P(+|x) > P(-|x) \\ - & \text{if } P(+|x) < P(-|x) \end{cases}$$

- This is called the **Maximum A Posteriori** decision rule.

$$P(+|x) > P(-|x)$$

- This boils down to (after plugging in)

$$\frac{p(x|+) \times P(+)}{p(x)} > \frac{p(x|-) \times P(-)}{p(x)}$$

$$\frac{p(x|+)}{p(x|-)} > \frac{P(-)}{P(+)}$$

- Here, we have just two values that Y can take, so the decision rule is

$$P(+|x) > P(-|x) \Rightarrow P(+|x) > 1 - P(+|x) \Rightarrow P(+|x) > 1/2$$

MAP Decision Rule Minimizes Test Error

- It turns out that the MAP decision rule is *Bayes optimal*, it minimizes overall test error. We do not provide a formal proof, but the intuition is given
- What is the error that we make incorporating the information contained in x?

$$P(error|x) = \min\{P(+|x), P(-|x)\}$$

- The total error over all x would be:

$$P(error) = \int_0^\infty \overbrace{p(error, x)}^{\text{joint}} dx = \int_0^\infty P(error|x)p(x)dx$$

- MAP decision rule minimizes this error, because for every value of x, we choose

$$\min\{P(+|x), P(-|x)\}$$

Plan

Classification - Types of Errors

- In regression, we would evaluate the performance of a predictive model $\hat{f}$ by studying MSE on a validation set.

$$MSE = \frac{1}{n}\sum_{j}^{n}(y_j - f(\hat{x}_{ij}))^2$$

- For numeric variables, we can over-estimate or underestimate. We can tweak our objective function used to estimate the coefficients to penalize over- versus underestimates.

Test and Training Error

- We have argued that the intuitive Maximum A Posteriori Decision rule is *optimal*, i.e. it reduces the overall training error.
- Accuracy is computed as the count of the correct assignments relative to the validation set size N, formally

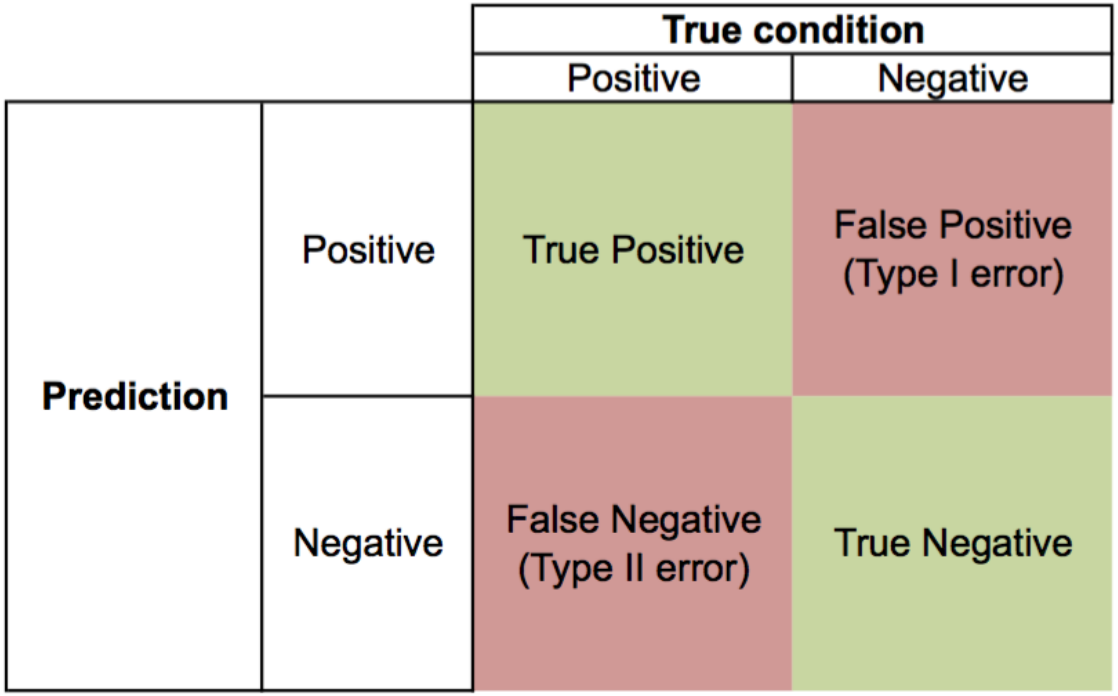
$$\frac{1}{N}\sum_{j=1}^{N} I(\hat{Y}_j = Y_j)$$

where MAP says

$$\hat{Y} = argmax_{y \in \mathcal{C}} \hat{P}(Y = y|X)$$

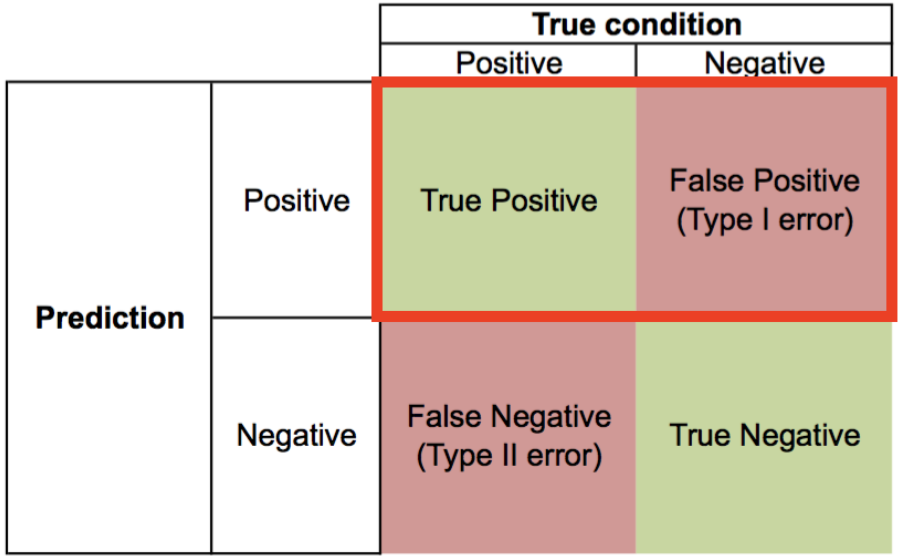
- As opposed to regression, we can better distinguish the types of errors we make.

Two Types of Errors



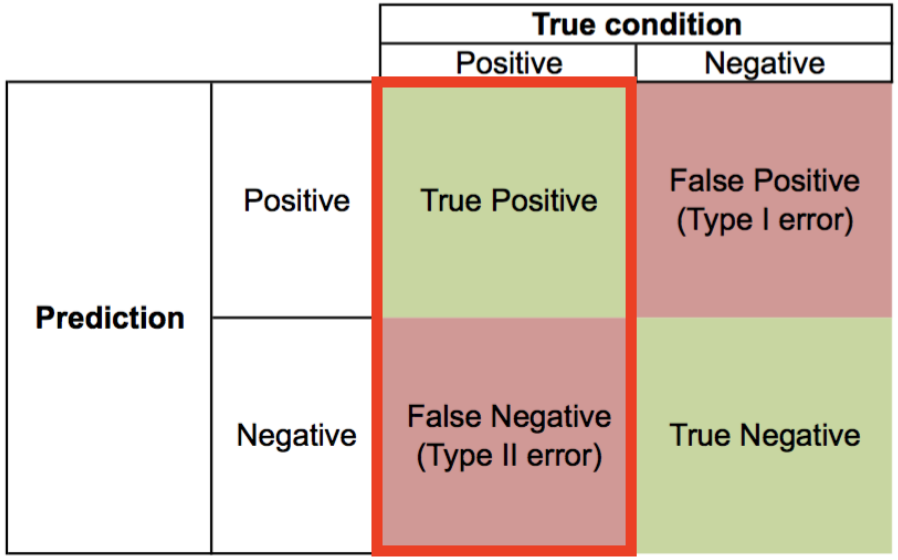
Important: the MAP decision rule minimizes the *overall error rate*. But this may come at the expense of high (low) type 1 versus low (high) type 2 error rates!

Precision, Recall and Accuracy



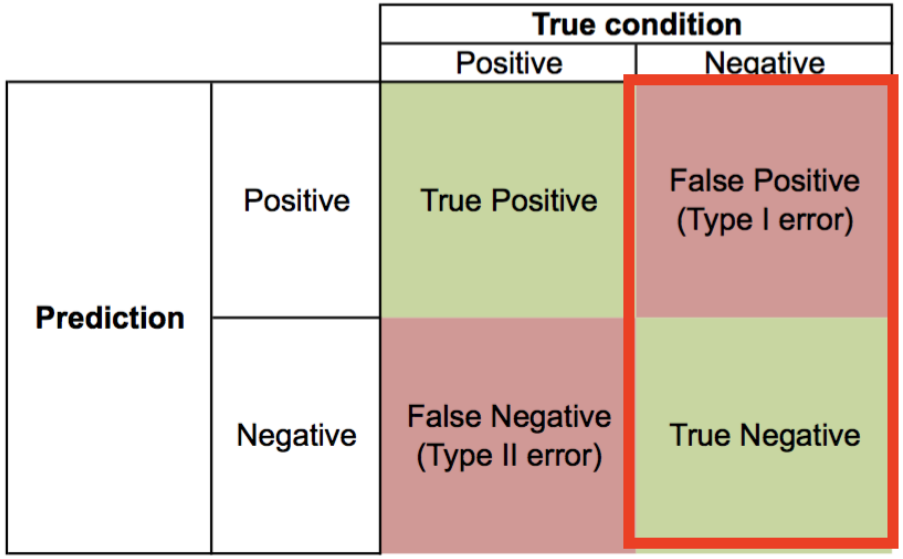
- $Precision = \frac{TruePositives}{TruePositive + FalsePositive}$

Precision, Recall and Accuracy



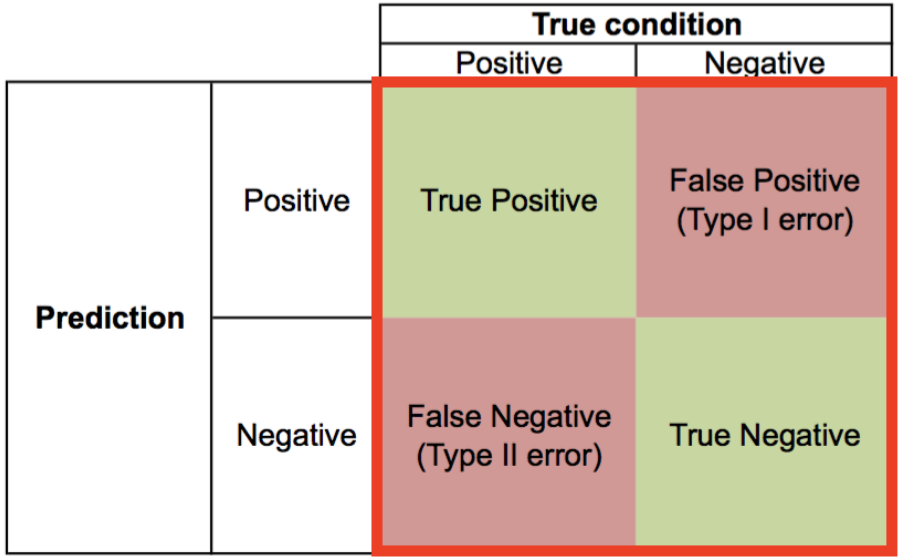
- $Precision = \frac{TruePositives}{TruePositive + FalsePositive}$
- $Recall = \frac{TruePositives}{TruePositive + FalseNegatives}$

Precision, Recall and Accuracy



- $Precision = \frac{TruePositives}{TruePositive + FalsePositive}$
- $Recall = \frac{TruePositives}{TruePositive + FalseNegatives}$
- $Specificity = \frac{TrueNegative}{TrueNegative + FalsePositive}$

Precision, Recall and Accuracy



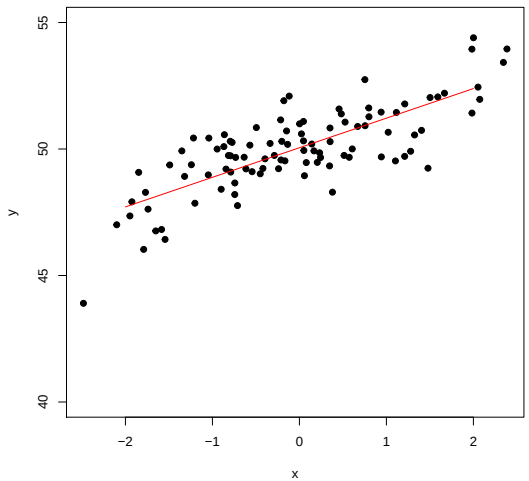
- $Precision = \frac{TruePositives}{TruePositive + FalsePositive}$
- $Recall = \frac{TruePositives}{TruePositive + FalseNegatives}$
- $Specificity = \frac{TrueNegative}{TrueNegative + FalsePositive}$
- $Accuracy = \frac{TruePositives + TrueNegative}{AllCases}$

Precision, Recall and Accuracy

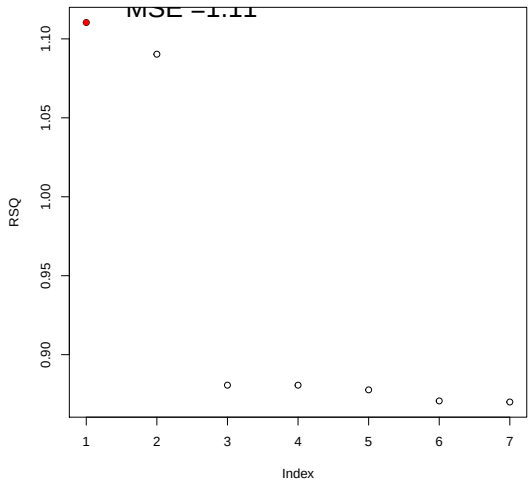
- The MAP decision rule minimizes overall test error, i.e. it maximizes Accuracy $\Rightarrow$ this could however have any implications regarding the distribution of errors across false positive or false negative
- As with numeric prediction, one central issue is that of over-fitting the data.

Training MSE versus test MSE Evolution

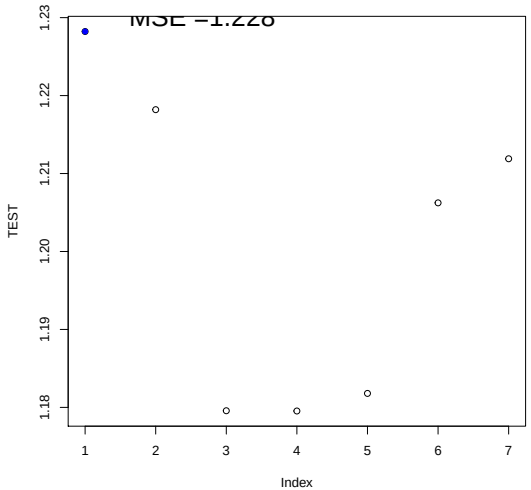
Polynomial of order 1



Training MSE



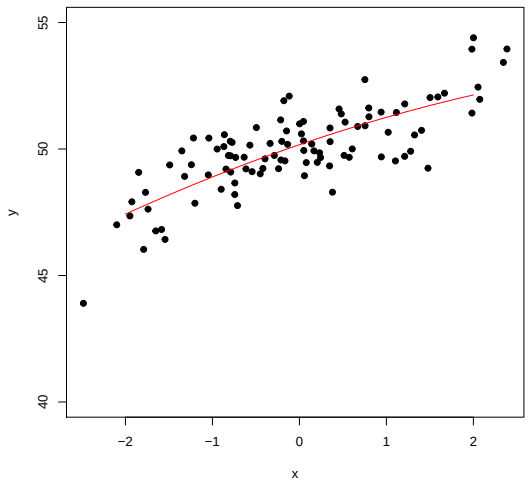
Test MSE



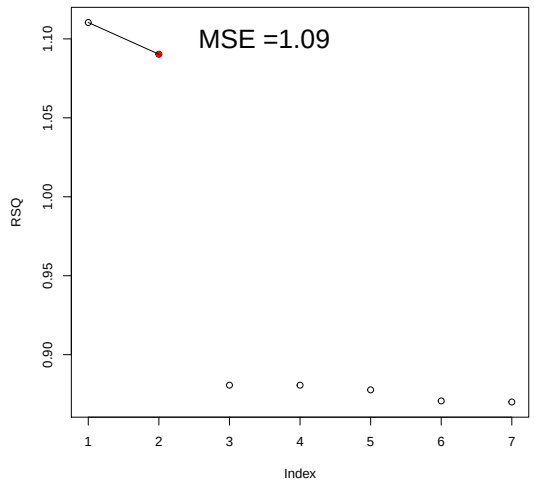
This is fitting a 1-th order polynomial to the scatterplot on the left. As we increase the order, the **training MSE** decreases monotonically, while the **test MSE** stops decreasing after a certain point. We are *overfitting* the data.

Training MSE versus test MSE Evolution

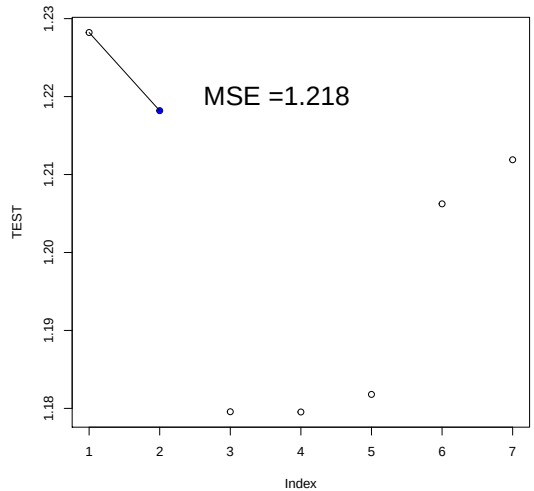
Polynomial of order 2



Training MSE



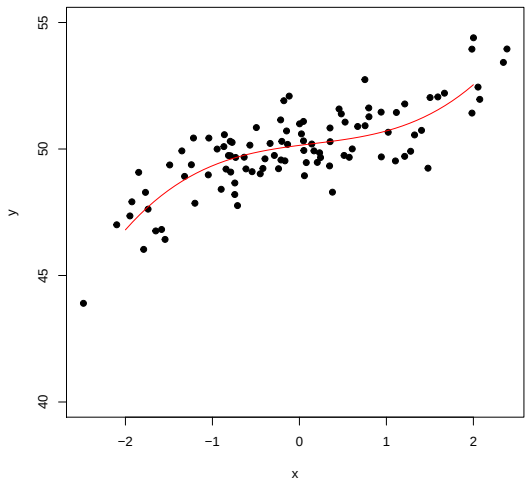
Test MSE



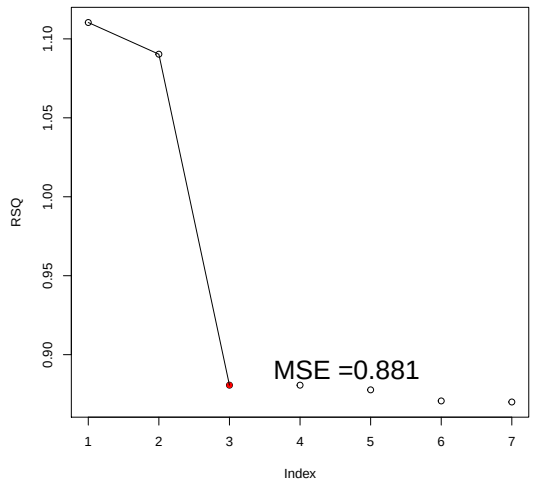
This is fitting a 2-th order polynomial to the scatterplot on the left. As we increase the order, the **training MSE** decreases monotonically, while the **test MSE** stops decreasing after a certain point. We are *overfitting* the data.

Training MSE versus test MSE Evolution

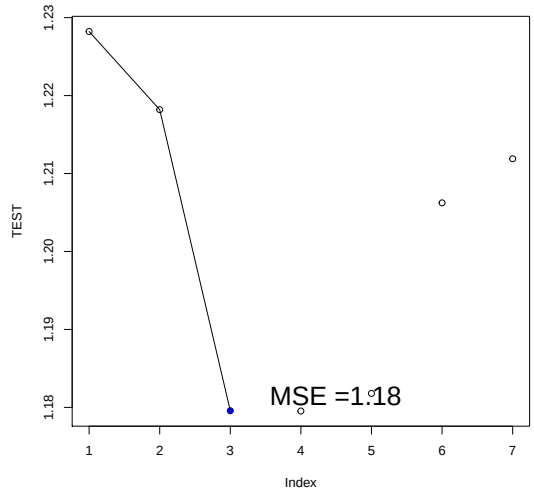
Polynomial of order 3



Training MSE



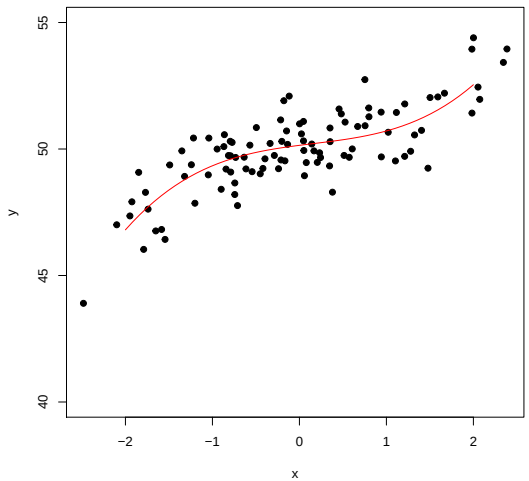
Test MSE



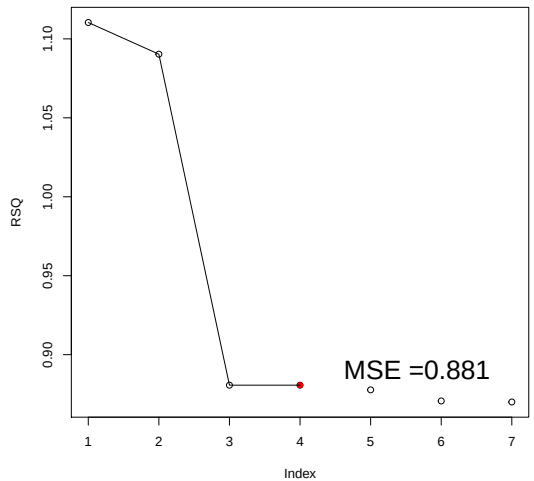
This is fitting a 3-th order polynomial to the scatterplot on the left. As we increase the order, the **training MSE** decreases monotonically, while the **test MSE** stops decreasing after a certain point. We are *overfitting* the data.

Training MSE versus test MSE Evolution

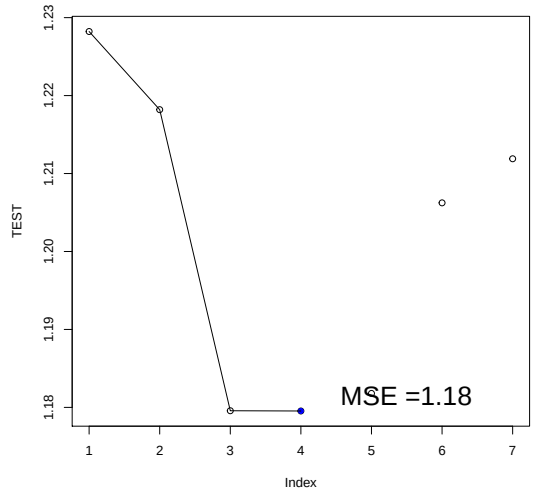
Polynomial of order 4



Training MSE



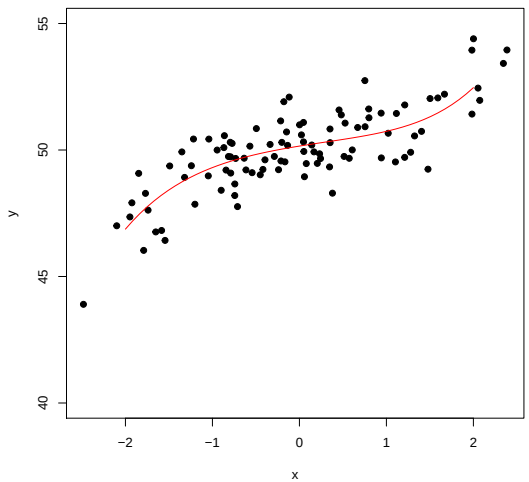
Test MSE



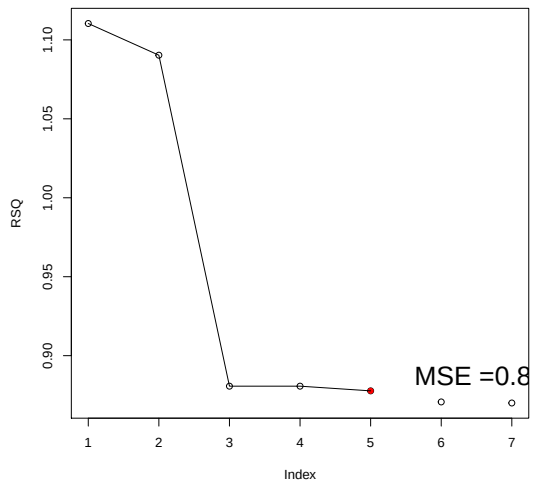
This is fitting a 4-th order polynomial to the scatterplot on the left. As we increase the order, the **training MSE** decreases monotonically, while the **test MSE** stops decreasing after a certain point. We are *overfitting* the data.

Training MSE versus test MSE Evolution

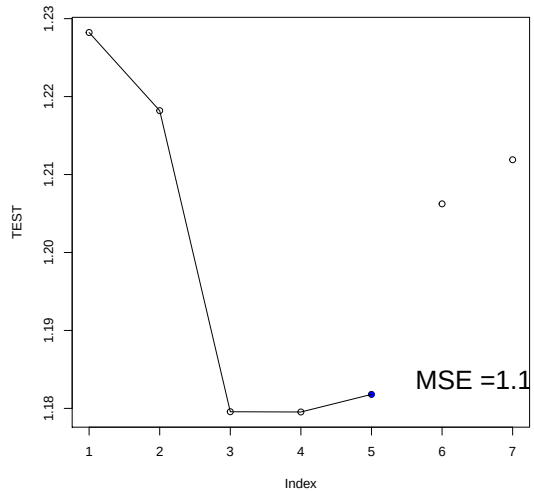
Polynomial of order 5



Training MSE



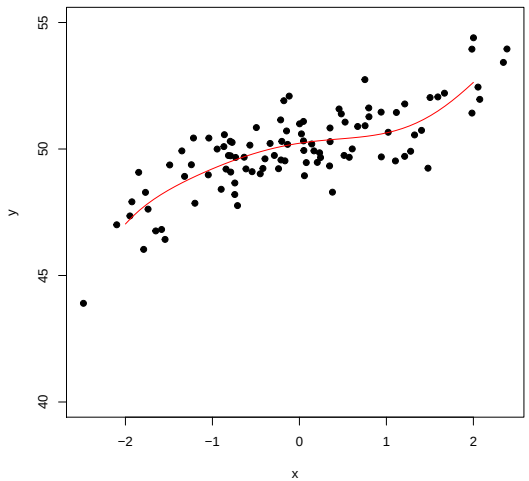
Test MSE



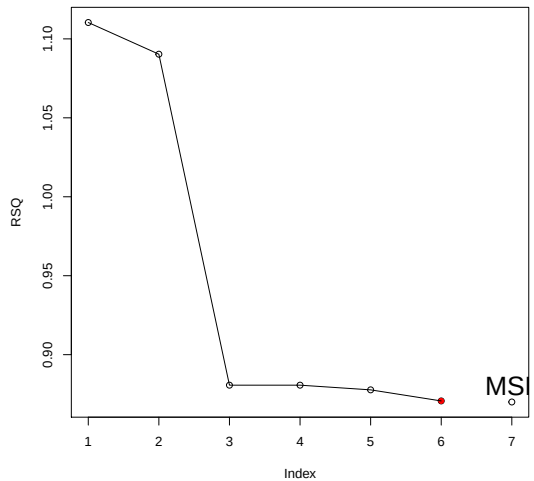
This is fitting a 5-th order polynomial to the scatterplot on the left. As we increase the order, the **training MSE** decreases monotonically, while the **test MSE** stops decreasing after a certain point. We are *overfitting* the data.

Training MSE versus test MSE Evolution

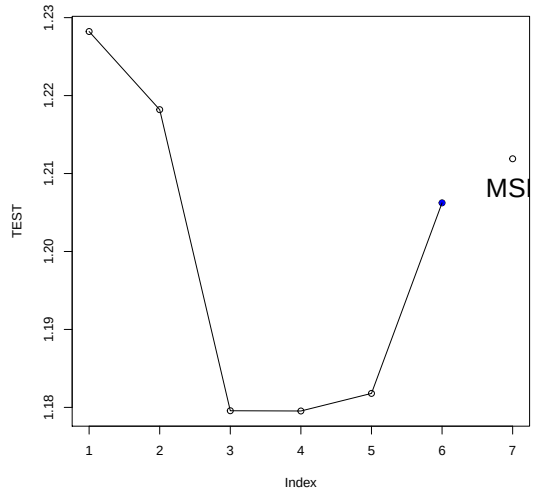
Polynomial of order 6



Training MSE



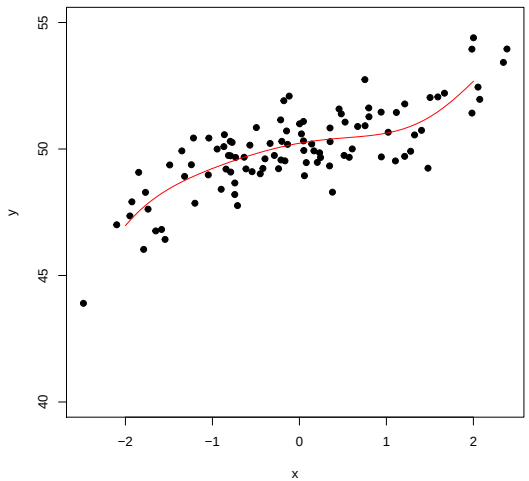
Test MSE



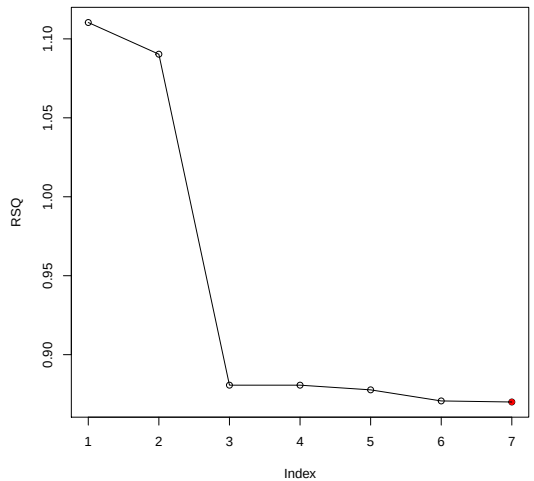
This is fitting a 6-th order polynomial to the scatterplot on the left. As we increase the order, the **training MSE** decreases monotonically, while the **test MSE** stops decreasing after a certain point. We are *overfitting* the data.

Training MSE versus test MSE Evolution

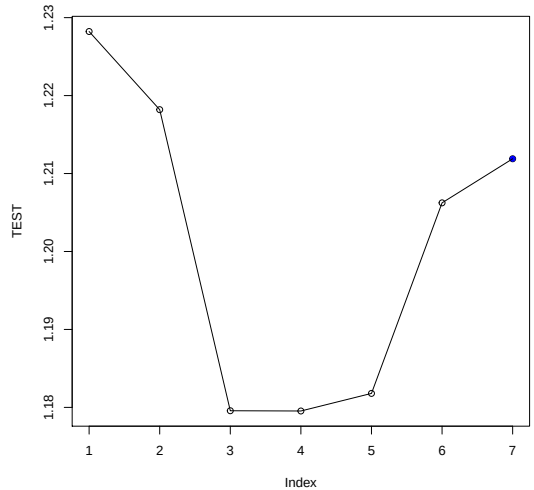
Polynomial of order 7



Training MSE



Test MSE



This is fitting a 7-th order polynomial to the scatterplot on the left. As we increase the order, the **training MSE** decreases monotonically, while the **test MSE** stops decreasing after a certain point. We are *overfitting* the data.

Overfitting...why does this happen?

- In the given example, we know that the minimal attainable test MSE $=1$, since the irreducible error has variance 1, given $\epsilon \sim N(0, 1)$.
- Fitting ever more complicated polynomials, while ignoring the true model implies, that the estimated models are starting to **explain the noise** contained in ϵ.
- From Econometrics: including *irrelevant variables* (that is those with coefficients ≈ 0) does not result in biased point estimates, but the resulting estimators are not *efficient*, i.e. OLS is not BLUE.
- Since the noise is randomly drawn and thus, the noise in the training set is **independent** from the noise in the test set, the performance of the fit estimated from the data in the training set will become worse and worse

Plan

The General Classification Problem

- A qualitative variable takes a value in set $\mathcal{C}$, for example Teaching Feedback $\in \{$Positive, Negative, Neutral$\}$, or, binary, Email $\in \{$Spam, No Spam$\}$.
- We will denote $|\mathcal{C}| = c$ the number of different categories.
- We still observe a feature vector X and a responsive variable Y that takes on values from the set $\mathcal{C}$
- The MAP Decision rule says that we should assign

$$\hat{Y} = argmax_{y \in \mathcal{C}} \hat{P}(Y = y|X)$$

- As we saw, in the binary case, where $c = 2$, this boils down to assigning an observation to $Y = 1$ if $\hat{P}(Y = 1|X) > .5$

Getting a $\hat{P}(Y = y|X)$

There are different ways to arrive at $\hat{P}(Y = y|X)$.

- In this section we will present three methods for obtaining a $\hat{P}(Y = y|X)$.
- They belong to two distinct classes of estimators
  1. Discriminative
  2. Generative

Difference between Generative vs Discriminative

The classification problem

$$\hat{Y} = argmax_{y \in \mathcal{C}} \hat{P}(Y = y|X)$$

A conditional probability is defined as $P(Y|X) = \frac{P(X,Y)}{P(X)}$

1. A **discriminative model** tries to learn the distribution of $P(Y|X)$
2. A **generative model** tries to learn the data generating distribution $P(X, Y)$

We will start by discussing logistic regression, which is explicitly discriminative, we then discuss k-Nearest Neighbours as an intuitive model, that turns out to be discriminative; lastly, we discuss Naive Bayes as an example of an explicitly generative model.

Logistic Regression

- Logistic regression takes the form

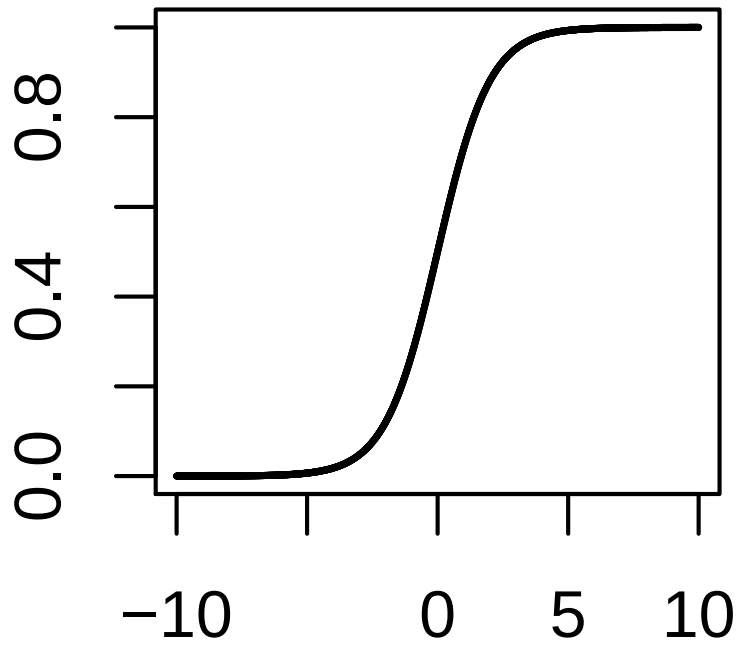
$$P(Y = c|X) = h(\beta_0 + \sum_{k=1}^{p} \beta_k X_k)$$

- where $h(z)$ is the sigmoid function.

$$h(z) = \frac{e^z}{1 + e^z}$$

- Note as $z \to -\infty$,
- $h(z) \to 0$, $z = 0, h(z) = 1/2$ and
- $z \to \infty, h(z) \to 1$.

Logistic Regression



Substituting:

$$P(Y = 1|X) = \frac{e^{\beta_0 + \sum_{k=1}^{p} \beta_k X_k}}{1 + e^{\beta_0 + \sum_{k=1}^{p} \beta_k X_k}}$$

Logistic Regression...is linear regression in disguise

Suppose you call $P(Y = 1|X) = p(X)$, then the logistic function becomes.

$$p(X\beta) = \frac{e^{\beta_0 + \sum_{k=1}^{p} \beta_k X_k}}{1 + e^{\beta_0 + \sum_{k=1}^{p} \beta_k X_k}} = \frac{e^{X\beta}}{1 + e^{X\beta}}$$

You can rearrange this as

$$\frac{p(X\beta)}{1 - p(X\beta)} = e^{\beta_0 + \sum_{k=1}^{p} \beta_k X_k} = e^{X\beta}$$

This looks almost like a linear model... the LHS has an intuitive explanation. In the numerator is $P(Y = 1|X)$, while the denominator is $P(Y = 0|X)$. This is called the "odds ratio".
Taking logs:

$$\log(\frac{p(X\beta)}{1 - p(X\beta)}) = X\beta$$

Estimating Logistic Regression using MLE

The coefficient vector β is unknown. We will estimate it by maximizing the likelihood function $\rightarrow$ **Maximum Likelihood**.
We assume that the individual pairs of observations are *independent* from one another.

$$\mathcal{L}(\beta|X) = \prod_{i:y_i=1} p(x_i) \prod_{j:y_j=0} (1 - p(x_j))$$

This is the joint likelihood of observing, out of a sample of N observations, a subset I with $y_i = 1$ and a complementary subset J with $y_j = 0$:

Estimating Logistic Regression using MLE

We can rewrite in the binary case since $y_i \in \{0, 1\}$, so the likelihood of an individual observation y_i

$$p(x_i'\beta)^{y_i} \times (1 - p(x_i'\beta))^{1-y_i}$$

The joint likelihood becomes:

$$\mathcal{L}(\beta|X) = \prod_{i=1}^{n} p(x_i'\beta)^{y_i}(1 - p(x_i'\beta))^{1-y_i}$$

Rewriting the Log Likelihood Function

Taking logs...

$$log(\mathcal{L}(\beta|X)) = \sum_{i=1}^{n} y_i \log(p(x_i'\beta)) + (1 - y_i) \log((1 - p(x_i'\beta)))$$

You want to find a vector β that maximizes the above expression. Since the function is concave, a maximum will statisfy a first order condition:

$$\frac{\partial log(\mathcal{L}(\beta|X))}{\partial \beta} = \sum_{i=1}^{n} (y_i - p(x_i'\beta))x_i = 0$$

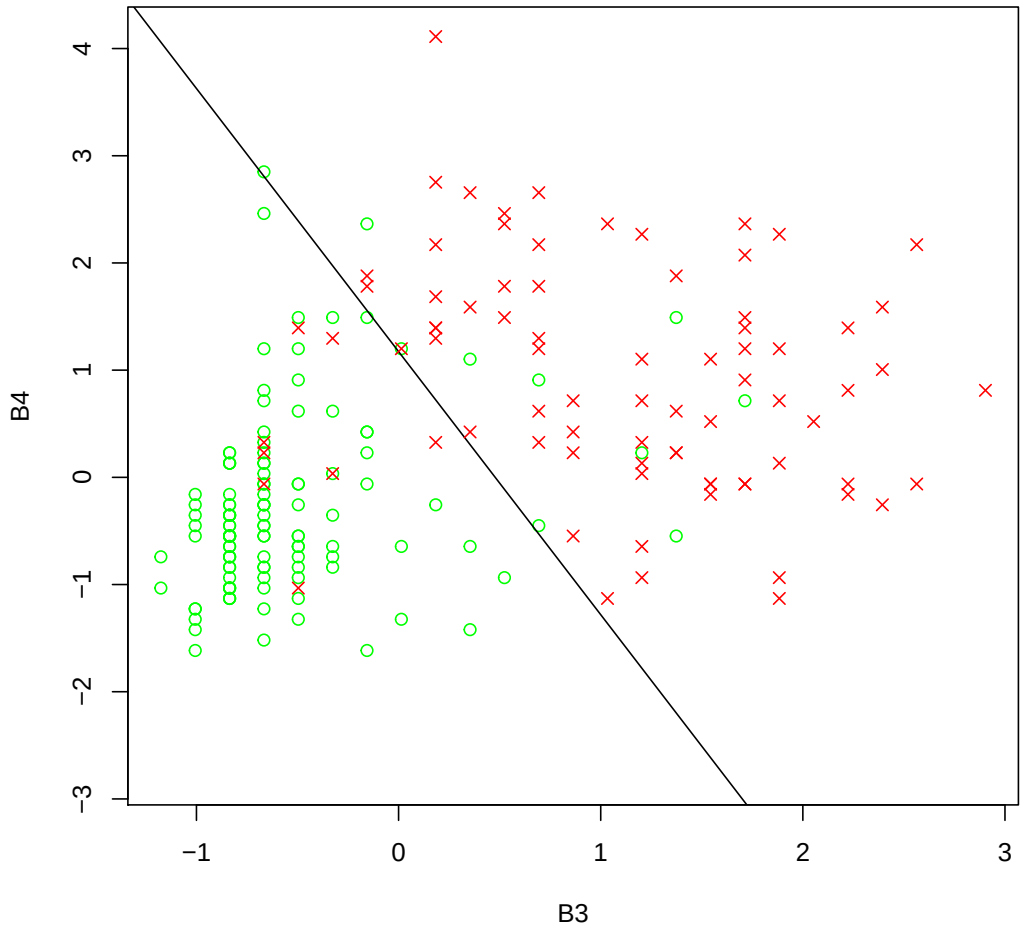
The reason why the FOC is relatively easy to derive, is because the derivative of a logistic is easy...

$$h(z) = \frac{e^z}{1 + e^z}$$

One can show that

$$h'(z) = h(z)(1 - h(z))$$

A simple example: Logistic Regression



Here, we fit a logistic regression with two variables and an intercept on a dummy variable.

A simple example: Logistic Regression

The output in R looks as follows:

```
summary(glm.fit)
##
## Call:
## glm(formula = Forested ~ B3 + B4, family = binomial(link = logit),
##     data = DF.plot)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.600  -0.330   0.195   0.324   2.715
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)    1.146      0.289    3.97  7.3e-05 ***
## B3            -2.398      0.348   -6.90  5.3e-12 ***
## B4            -0.977      0.239   -4.08  4.5e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 269.20  on 199  degrees of freedom
## Residual deviance: 107.48  on 197  degrees of freedom
## AIC: 113.5
##
## Number of Fisher Scoring iterations: 6
```

A simple example: Logistic Regression and MAP Rule

- The Maximum Likelihood is estimated using the log-odds transformation, so we have

$$\log(\frac{p(X\beta)}{1-p(X\beta)}) = \hat{\beta}_0 + X_1\hat{\beta}_1 + X_2\hat{\beta}_2$$

- With $\hat{\beta}_0 = 1.146$, $\hat{\beta}_1 = -2.398$ and $\hat{\beta}_2 = -0.997$.
- We minimize the Bayes Error rate if we follow the MAP Decision Rule!
- MAP Decision rule. Set $y_i = 1$ if $\hat{P}(y_i = 1|x_i) > 1/2$
- Now $\log(.5/(1-.5)) = 0$, so the MAP decision rule translates into

$$\hat{\beta}_0 + X_1\hat{\beta}_1 + X_2\hat{\beta}_2 > 0$$

A simple example: Logistic Regression and Decision Boundary

We have seen that the MAP Decision Rule translates to

$$\hat{\beta}_0 + X_1\hat{\beta}_1 + X_2\hat{\beta}_2 > 0$$

So we can rewrite this as an equation

$$X_2 > -\frac{\hat{\beta}_0}{\hat{\beta}_2} - X_1\frac{\hat{\beta}_1}{\hat{\beta}_2}$$

Which in this case is:

$$X_2 > \frac{1.146}{0.997} + \frac{-2.398}{0.977}X_1$$

which naturally we can plot, since its a straight line with intercept.

An Example with Real Data: Mortgage Applications

- We start with an example where $|\mathcal{C}| = c = 2$, i.e. a binary example. For this purpose, we use data from the US Home Mortgage Disclosure Act (HMDA).
- The HMDA requires certain financial institutions to provide data on all mortgage applications and decisions on these applications, along with some characteristics.
- Pretty big database, in 2012, there were 7,400 institutions that reported a total of 18.7 million HMDA records.
- We want to see which patterns predict, whether an application is $\{\text{Rejected}, \text{Not Rejected}\}$. This can be expressed as a dummy variable, where

$$Y = \begin{cases} 1 & \text{if Application rejected} \\ 0 & \text{if Application granted} \end{cases}$$

Data available here : `https://www.ffiec.gov/hmda/`

Summary statistics

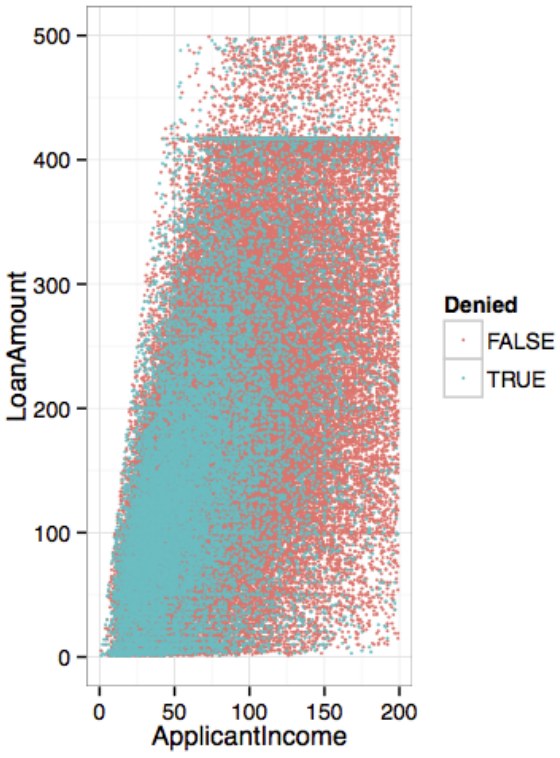
```
head(MORTGAGE[,c("Denied","Leverage","Minority","ApplicantIncome","LoanAmount","Female",
                 "LoanPurpose","Occupancy"),with=F])
```

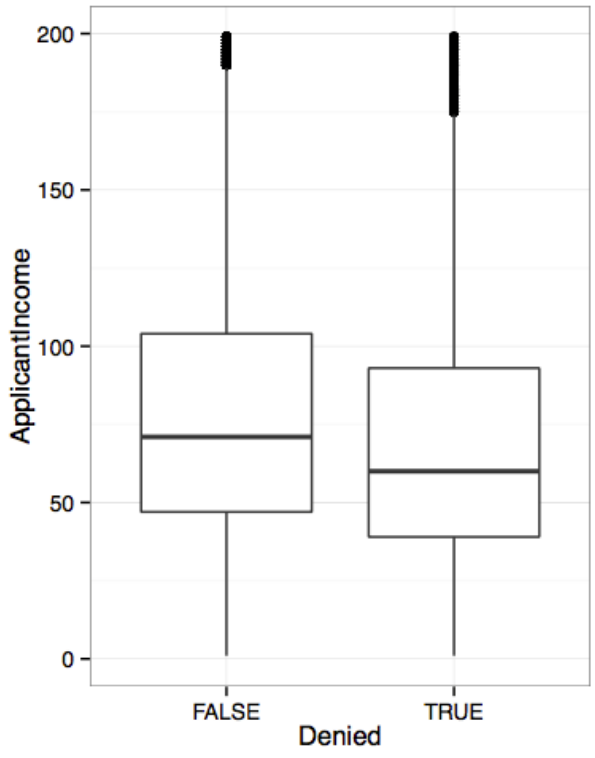
```
##    Denied Leverage Minority ApplicantIncome LoanAmount Female   LoanPurpose
## 1:  FALSE     2.54    FALSE              61        155      0   Refinancing
## 2:  FALSE     1.50    FALSE              40         60      1 Home purchase
## 3:  FALSE     2.29    FALSE             136        311      0   Refinancing
## 4:  FALSE     1.39    FALSE              31         43      1 Home purchase
## 5:  FALSE     3.99    FALSE              90        359      1   Refinancing
## 6:  FALSE     5.62    FALSE              53        298      0   Refinancing
##         Occupancy
## 1: Owner-occupied
## 2: Owner-occupied
## 3: Owner-occupied
## 4: Owner-occupied
## 5: Owner-occupied
## 6: Owner-occupied
```

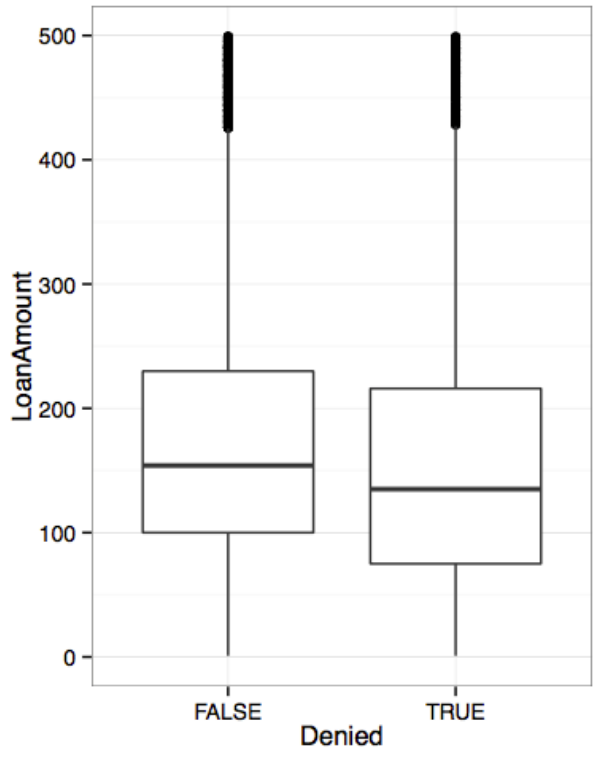
```
summary(MORTGAGE[,c("Denied","Leverage","Minority","ApplicantIncome","LoanAmount","Female",
                    "LoanPurpose","Occupancy"),with=F])
```

```
##    Denied           Leverage      Minority       ApplicantIncome   LoanAmount
##  Mode :logical   Min.   :0.01   Mode :logical   Min.   :  1     Min.   :  1
##  FALSE:102481    1st Qu.:1.51   FALSE:98041     1st Qu.: 46     1st Qu.: 97
##  TRUE :16354     Median :2.27   TRUE :20794     Median : 70     Median :151
##  NA's :0         Mean   :2.43   NA's :0         Mean   : 78     Mean   :170
##                  3rd Qu.:3.18                   3rd Qu.:103     3rd Qu.:228
##                  Max.   :9.96                   Max.   :199     Max.   :499
##      Female                LoanPurpose                  Occupancy
##  Min.   :0.000   Home improvement: 5912   Non Owner-occupied:  8002
##  1st Qu.:0.000   Home purchase   :43119   Owner-occupied    :110833
##  Median :0.000   Refinancing     :69804
##  Mean   :0.298
##  3rd Qu.:1.000
##  Max.   :1.000
```

Some graphs...


LoanAmount
ApplicantIncome
Denied
FALSE
TRUE


ApplicantIncome
FALSE
TRUE
Denied


LoanAmount
FALSE
TRUE
Denied

Estimating Logistic Regression using MLE

```
glm.fit<-glm(Denied ~  Leverage + Minority + ApplicantIncome + Female + Occupancy + LoanPurpose,
             data=MORTGAGE, family=binomial(link=logit))
summary(glm.fit)
```

```
##
## Call:
## glm(formula = Denied ~ Leverage + Minority + ApplicantIncome +
##     Female + Occupancy + LoanPurpose, family = binomial(link = logit),
##     data = MORTGAGE)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.394  -0.570  -0.499  -0.427   2.529
##
## Coefficients:
##                            Estimate Std. Error z value Pr(>|z|)
## (Intercept)               -0.203883   0.046629   -4.37  1.2e-05 ***
## Leverage                   0.048997   0.006889    7.11  1.1e-12 ***
## MinorityTRUE               0.332573   0.021017   15.82  < 2e-16 ***
## ApplicantIncome           -0.006041   0.000248  -24.36  < 2e-16 ***
## Female                     0.115838   0.018498    6.26  3.8e-10 ***
## OccupancyOwner-occupied   -0.520638   0.031582  -16.49  < 2e-16 ***
## LoanPurposeHome purchase -1.260043   0.035230  -35.77  < 2e-16 ***
## LoanPurposeRefinancing    -0.838062   0.033596  -24.94  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 95215  on 118834  degrees of freedom
## Residual deviance: 92564  on 118827  degrees of freedom
## AIC: 92580
##
## Number of Fisher Scoring iterations: 4
```

Interpreting the Output...

- The reported coefficients tell you what is the marginal effect of a change in some X_i on the log-odds ratio.
- Hence, you can interpret the signs, but the coefficients are not the marginal effects in terms of probabilities.
- The marginal effects on the probabilities are **not constant**.
- The effect of the odds of a 1-unit increase in Leverage is $exp(0.048) = 1.048$, meaning an odds increase by around 4.8%.
- Note: For small x, $e^x \approx 1 + x$.
- Lets look at the range of the predicted probabilities for a more saturated model

```
predpr <- predict(glm.fit,type=c("response"))
summary(predpr)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.040   0.103   0.127   0.138   0.159   0.621
```

MAP Decision Rule and Error Types

- Suppose you were to suggest to *only screen bad loan applications* intensively, i.e. those whose probability of being rejected are above a threshold $\bar{c}$.
- MAP Decision rule says that the overall error rate is minimal, in case you set $\bar{c} = 1/2$.
- Lets see how we would do in this case.

Classification and MAP Rule: How are we doing?

MAP Decision rule says that the overall error rate is minimal, in case you set $\bar{c} = 1/2$.

```
Denied=as.character(MORTGAGE[test]$Deniedfactor)
glm.probs=predict(glm.fit,MORTGAGE[test],type="response")
glm.pred=rep("Non intense check",length(glm.probs))
glm.pred[glm.probs>.5]="Intense check"
addmargins(table(glm.pred,Denied))
```

```
##                    Denied
## glm.pred            Denied Granted  Sum
##   Intense check          6       7   13
##   Non intense check    123     864  987
##   Sum                  129     871 1000
```

How do we do?

- Accuracy 6+864=870/1000 or almost 87% correctly classified
- Precision $= 6/(6+7) = 46\%$.
- Recall $= 6/(6+123) = 4\%$.
- Specificity $= 864/871 = 99\%$.

We get really high precision, because most loan applications are granted!

Not following MAP Rule...

Suppose you set $\bar{c} = 0.2$, out of our test sample of 1000 loan applications...

```
Denied=as.character(MORTGAGE[test]$Deniedfactor)
glm.probs=predict(glm.fit,MORTGAGE[test],type="response")
glm.pred=rep("Non intense check",length(glm.probs))
glm.pred[glm.probs>.2]="Intense check"
addmargins(table(glm.pred,Denied))
```

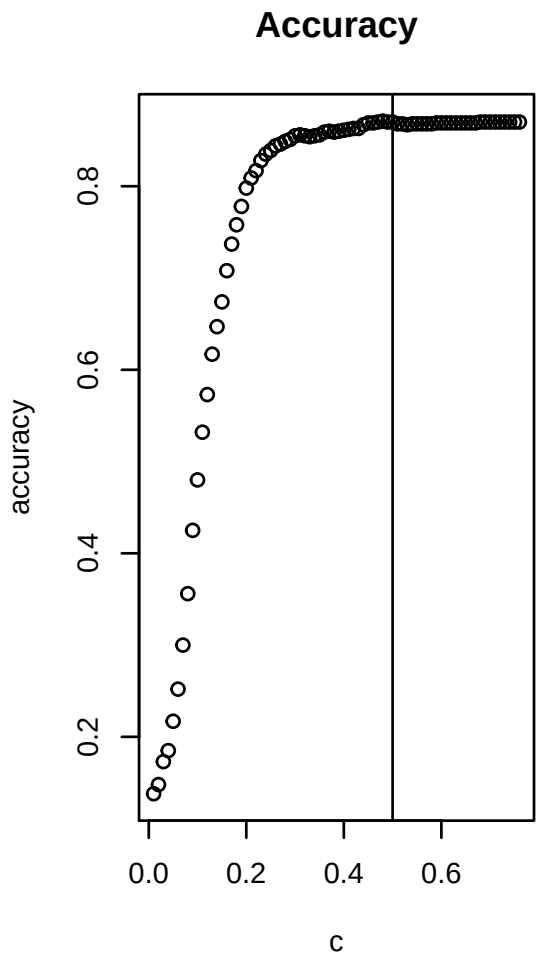
```
##                     Denied
## glm.pred             Denied Granted  Sum
##   Intense check         46     119  165
##   Non intense check     83     752  835
##   Sum                  129     871 1000
```

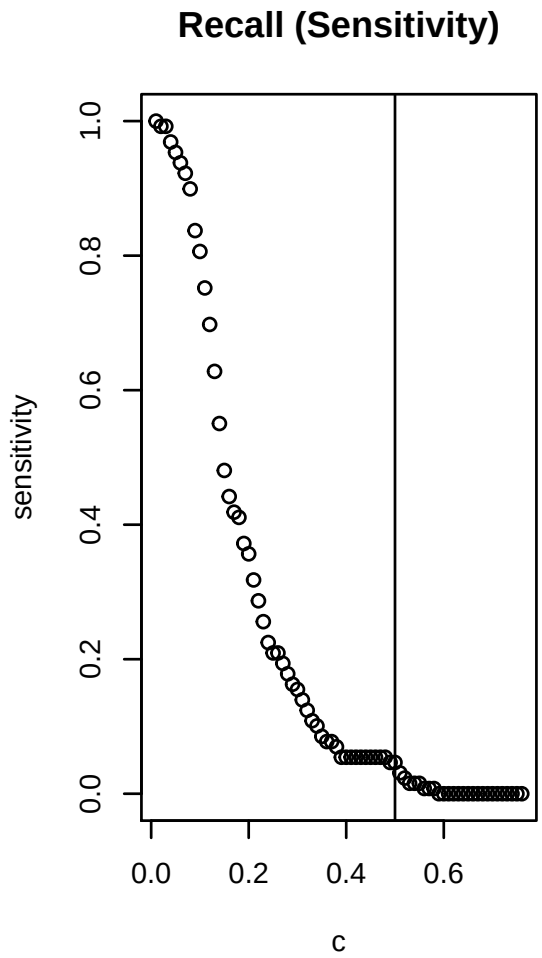
How do we do?

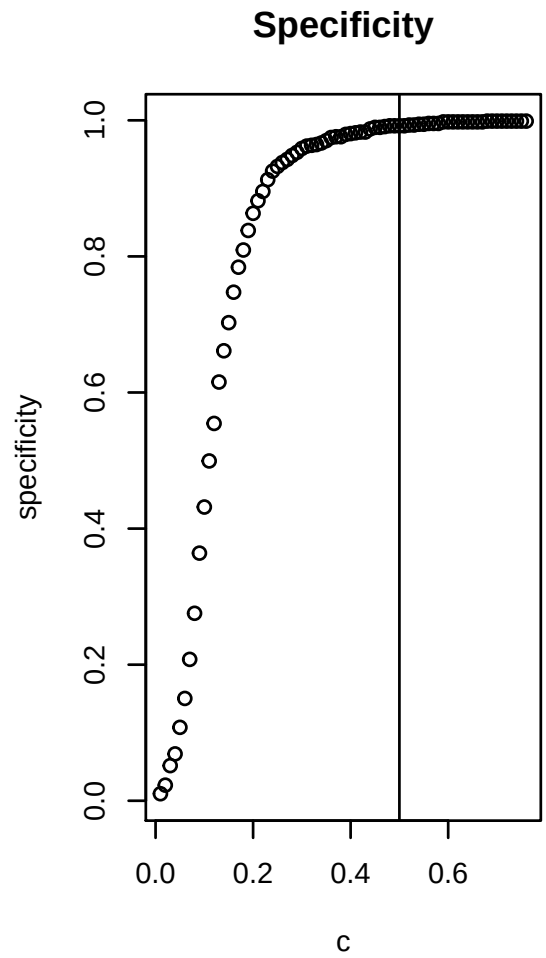
- Accuracy = (46+752) /1000 or almost 80% correctly classified
- Precision = 46 /(46+119) = 28%
- Recall = 46/(46+83) = 36%
- Specificity = 752/871 = 86%.

We are trading off true positives with true negatives.

Visualizing Accuracy, Recall and Specificty Tradeoffs







Trade off between Sensitivity and Specificity

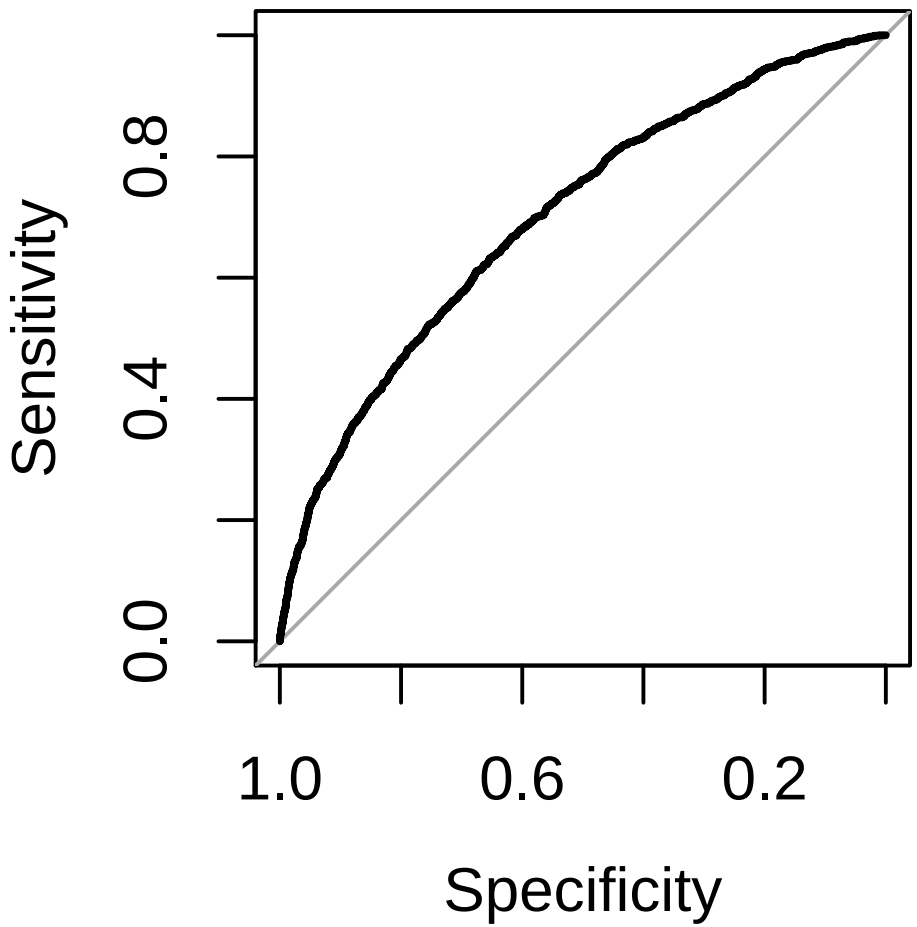
- As we increase $\bar{c}$, there is a trade off between the type 1 and type 2 errors that occur.
- For very low $\bar{c}$, a lot of loans are assigned to be intensively checked, resulting in many false positives (loans that would not have been denied being intensively checked), but relatively few false negatives - high sensitivity and low specificity.
- As we increase $\bar{c}$, fewer loans are intensively checked; this reduces the false positive cases but increases the false negative cases - low senitivity and high specificity.
- Overall, since most loans are granted (87.1%), the overall increase in accuracy as we increas $\bar{c}$ is driven by specificity.

ROC Curve to Visualize Trade Off between Sensitivity and Specificity

- ROC curve is a popular graphic for simultaneously displaying the two types of errors for all possible thresholds.
- It comes from communications theory and stands for "receiver operating characteristics".
- It plots Specificity against Sensitivity as we vary $\bar{c}$ (without showing $\bar{c}$)
- The ideal ROC curve is in the top left corner (100 % specificity and 100% sensitivity)
- The 45 degree line is the classifier that assigns observations to classes in a random fashion (e.g. a coin toss)
- The overall performance of a classifier, summarized over all possible thresholds, is given by the area under the (ROC) curve (AUC).

ROC Curve in our example

```
##
## Call:
## roc.formula(formula = MORTGAGE[training]$Denied ~ predpr)
##
## Data: predpr in 8622 controls (MORTGAGE[training]$Denied FALSE) < 1378 cases (MORTGAGE[training]$Denied TRUE).
## Area under the curve: 0.699
```



Choice of Decision Rule depends on cost

- The aim of this exercise was to highlight that choice of decision criterion, i.e. cutoff $\bar{c}$ need not be dictated by the MAP decision rule that guarantees, on average, best prediction performance.
- The question really is, what the associated costs are for having many tue positives or true negatives.
- Optimal choice of $\bar{c}$ would take into account the potentially different costs and may give solutions that are far away from $1/2$.

Model Selection Approaches

- In most machine learning applications, we are worried about overfitting.
- Overfitting can result from incorporating information contained in features that may explain - by chance - some of the variation in the training set, but perform poorly in the validation set.
- Shrinkage methods like Lasso can be used to remove features X that do not have a lot of information content in the validation sample
- Lasso for logistic regression would maximize Log Likelihood minus a penality:

$$\sum_{i=1}^{n} y_i \log(p(x_i'\beta)) + (1 - y_i) \log((1 - p(x_i'\beta))) - \lambda \sum_{j=1}^{p} |\beta_j|$$

An Example: Spam Data

- Most spam detection models use a combination of numeric features derived from the data as well as indicator and counts of word features.
- Numeric features are measures like the number or share of all caps words, length of uninterruted punctuations, sentence length,... you name it.
- Dummy variable or count variables of word feature counts.
- This exercise illustrates estimating logistic regression and performing Lasso to remove the feature space.

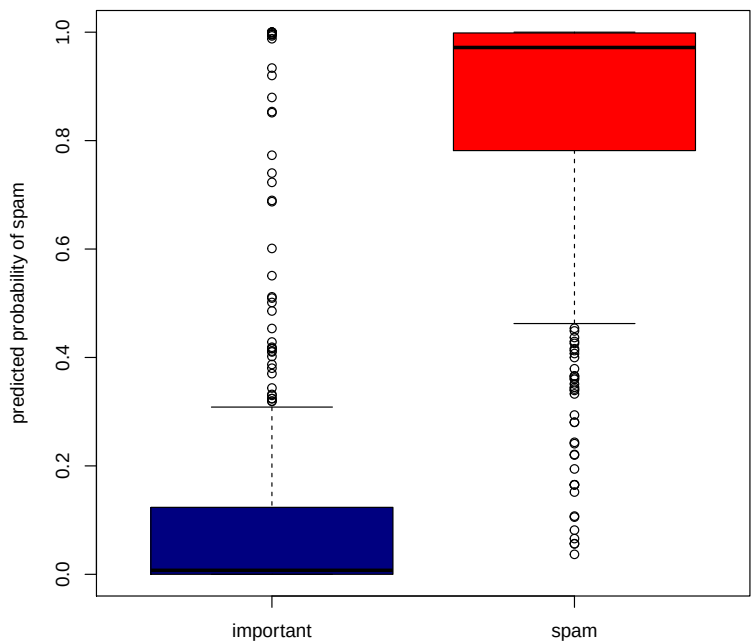
An Example: Spam Data

```
email <- read.csv("R/spam.csv")

email$spam <- factor(email$spam,levels=c(0,1),labels=c("important","spam"))

#features
names(email)
```

```
##  [1] "word_freq_make"             "word_freq_address"
##  [3] "word_freq_all"              "word_freq_3d"
##  [5] "word_freq_our"              "word_freq_over"
##  [7] "word_freq_remove"           "word_freq_internet"
##  [9] "word_freq_order"            "word_freq_mail"
## [11] "word_freq_receive"          "word_freq_will"
## [13] "word_freq_people"           "word_freq_report"
## [15] "word_freq_addresses"        "word_freq_free"
## [17] "word_freq_business"         "word_freq_email"
## [19] "word_freq_you"              "word_freq_credit"
## [21] "word_freq_your"             "word_freq_font"
## [23] "word_freq_000"              "word_freq_money"
## [25] "word_freq_hp"               "word_freq_hpl"
## [27] "word_freq_george"           "word_freq_650"
## [29] "word_freq_lab"              "word_freq_labs"
## [31] "word_freq_telnet"           "word_freq_857"
## [33] "word_freq_data"             "word_freq_415"
## [35] "word_freq_85"               "word_freq_technology"
## [37] "word_freq_1999"             "word_freq_parts"
## [39] "word_freq_pm"               "word_freq_direct"
## [41] "word_freq_cs"               "word_freq_meeting"
## [43] "word_freq_original"         "word_freq_project"
## [45] "word_freq_re"               "word_freq_edu"
## [47] "word_freq_table"            "word_freq_conference"
## [49] "char_freq_semicolon"        "char_freq_leftbrac"
## [51] "char_freq_leftsquarebrac"   "char_freq_exclaim"
## [53] "char_freq_dollar"           "char_freq_pound"
## [55] "capital_run_length_average" "capital_run_length_longest"
## [57] "capital_run_length_total"   "spam"
```

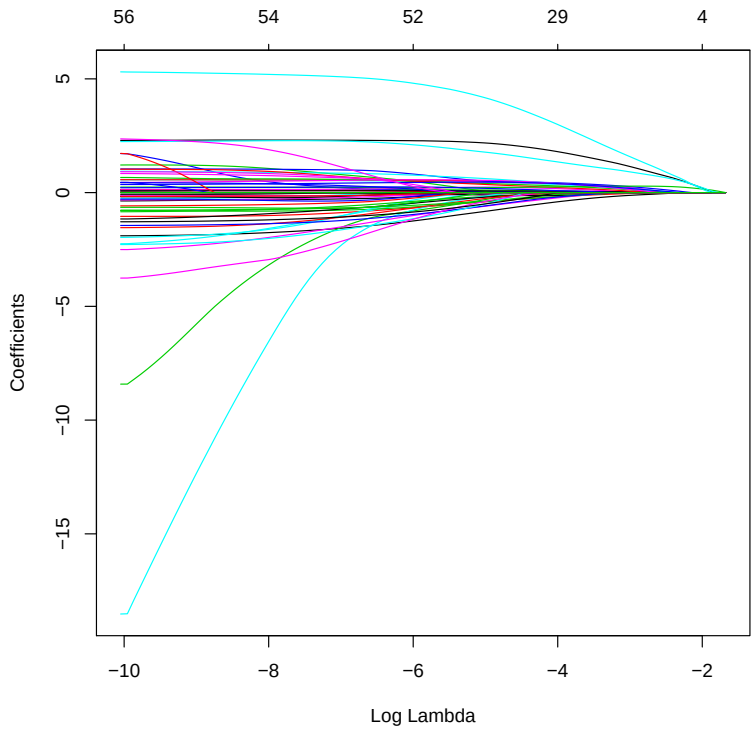
An Example: Validation Set

```
leaveout <- sample(1:nrow(email), 1000) ## sample 1000 random indices
# train the model WITHOUT these observations (-index removes those obs)
spamtrain <- glm(spam ~ ., data=email[-leaveout,], family='binomial')
# get the predicted probability of spam on the left out data
pspam <- predict(spamtrain, newdata=email[leaveout,], type="response")
# plot the OOS fit
plot(pspam ~ email$spam[leaveout],
        xlab="", ylab=c("predicted probability of spam"),
        col=c("navy","red"))
```



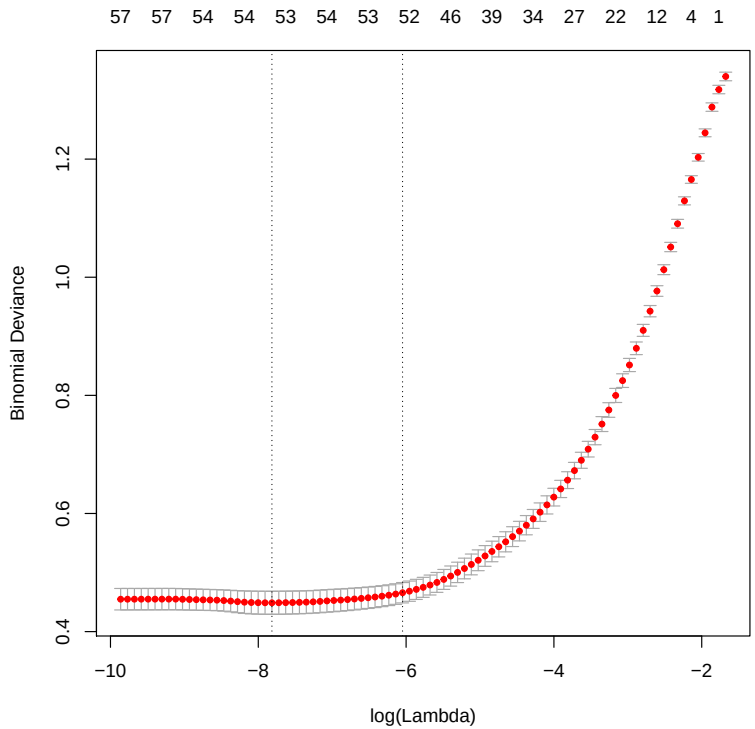
An Example: Regularized Regression

```
x<-model.matrix(spam~., data=email)[,-1]

lasso.mod=glmnet(x, as.factor(email$spam),alpha=1,standardize=TRUE,family='binomial')

plot(lasso.mod, xvar="lambda")
```



An Example: Cross Validation for Parameter Selection

```
cv.glmmod<-cv.glmnet(x,y=as.factor(email$spam),alpha=1,standardize=TRUE,family="binomial")

 plot(cv.glmmod)


cv.glmmod$lambda.min
## [1] 0.000403
```



Plan

Working with text features

- For short fragments, its most often useful to just dummify features
- Word counts at the sentence level do not make much sense
- I will show an example of logistic regression or multionomial logistic regression applied to a case of two categories categorizing short pieces of text.

Working with text features

```
library(data.table)
library(haven)
DTA<-data.table(read_dta(file="/Users/thiemo/Dropbox/Research/Blog/immigration/data/bes_f2f_original_v3.0.dta
DTA[, .N, by=A1][order(N,decreasing=TRUE)][1:10]
```

```
##                A1   N
##  1:  immigration 227
##  2:              131
##  3:  Immigration  52
##  4:      economy  44
##  5:          NHS  41
##  6: unemployment  40
##  7:    terrorism  24
##  8:  IMMIGRATION  20
##  9:   immigrants  20
## 10:  the economy  18
```

```
DTA[grep("immi",A1, ignore.case=TRUE)][nchar(A1)>10][1:3]$A1
```

```
## [1] "Immigration"         "too many immigrants" "Immigration"
## attr(,"label")
## [1] "A1: Most important issue"
## attr(,"format.stata")
## [1] "%244s"
```

Defining text features for predictive exercise

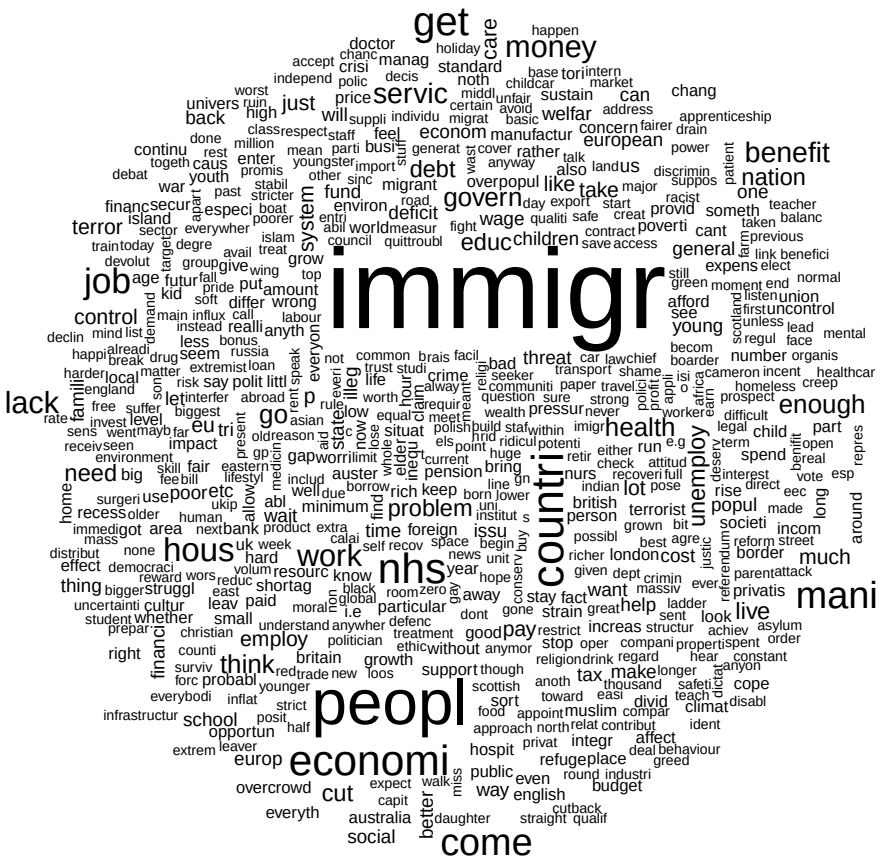
```
##use document scaling method to define features
DTA[, leaveeu := "remain"]
DTA[p02==1, leaveeu := "leave"]
DTA[p02<0 |p02>2, leaveeu := NA ]
DTA[, leaveeu := as.factor(leaveeu)]
DTA[, toomanyimmigrants := as.numeric(j05==1)]
DTA[j05<0 |j05>2, toomanyimmigrants := NA ]
DTA<-DTA[!is.na(leaveeu) & !is.na(toomanyimmigrants)]
table(DTA[ ,list(toomanyimmigrants, leaveeu)])
```

```
##                  leaveeu
## toomanyimmigrants leave remain
##                 0    60    478
##                 1   872    777
```

```
library(quanteda)
C<-corpus(DTA$A1)
##defining features that are informative about leaveeu decision using e.g. document scaling method
docnames(C)<-DTA$finalserialno
C.dfm<-dfm(C, tolower = TRUE, stem = TRUE, removeNumbers=TRUE,removePunct=TRUE, remove = stopwords("english"))
```

Single most important issue facing the UK in 2015

```
plot(C.dfm)
```



Bayes scoring to identify relevant features

```
##use document scaling method to define features

##using Bayes scores to identify particular word features associated with Brexit
summary(DTA$leaveeu)
```

```
##  leave remain
##    932   1255
```

```
ws2 <- textmodel(C.dfm, as.numeric(DTA$leaveeu), model="NB", smooth=0)

#Words associated with LEAVE
scores<-sort(log(ws2$PwGc[1,]/ws2$PwGc[2,]),decreasing=FALSE)
scores<-scores[names(topfeatures(C.dfm, n=250))]
scores<-scores[!is.infinite(scores)]
scores<-sort(scores)

head(scores)
```

```
##    societi      young      littl    minimum manufactur       deal
##      -1.95      -1.73      -1.73      -1.73      -1.73      -1.60
```

```
features<-names(c(head(scores,100),tail(scores, 100)))
features<-features[nchar(features)>3]
```

Bayes scoring to identify relevant features

```
##remove features not in the leave/ remain list
dim(C.dfm)
```

```
## [1] 2187 1653
```

```
C.subdfm<-dfm_select(C.dfm, features=features , selection="keep")

dim(C.subdfm)
```

```
## [1] 2187  171
```

```
##convert DFM into a simple matrix
MAT<-data.frame("leaveeu"=DTA$leaveeu, C.subdfm)
leaveout <- sample(1:nrow(MAT), 200) ## sample 500 random indices
# train the model WITHOUT these observations (-index removes those obs)
trained <- glm(leaveeu ~ ., data=MAT[-leaveout,], binomial(link=logit))
# get the predicted probability of spam on the left out data
glm.probs <- predict(trained, newdata=MAT[leaveout,], type="response")
# plot the OOS fit

glm.pred=rep("remain",length(glm.probs))
glm.pred[glm.probs>0.5]="leave"

table(glm.pred,MAT[leaveout,]$leaveeu)
```

```
##
## glm.pred leave remain
##   leave     42     95
##   remain    38     25
```

Simple MaxEnt Classification in RTextTools

```
set.seed(2016)
TRAINING<-data.table(read.csv(file="/Users/thiemo/Dropbox/Research/Matteo and Thiemo/senna/classification-tree

PERSON<-TRAINING[objecttype=="person" & label1!=""]
PERSON$label1 <- factor(PERSON$label1)
PERSON$label1num <- as.numeric(factor(PERSON$label1))

head(PERSON[label1 =="civilian",paste(verb,objectcleanpp,sep=" ")])
```

```
## [1] "abducted civilians village morian"
## [2] "killed civilians"
## [3] "kidnapped civilian dingdongpara"
## [4] "killed civilian who had reportedly functioned counter insurgent few years ago"
## [5] "killed civilian"
## [6] "abducted civilians"
```

An Example on how to do simple MaxEnt Classification

```
head(PERSON[label1 =="terrorist",paste(verb,objectcleanpp,sep=" ")])
```

```
## [1] "killed maoist cpi aoist cadre"
## [2] "raided maoist hideout village forest area abuj mad"
## [3] "surrendered maoists"
## [4] "killed terrorists united liberation front asom ulfa"
## [5] "injuring least militants"
## [6] "arrested cpi aoist cadres village"
```

An Example on how to do simple MaxEnt Classification

```
library(tm)
library(RTextTools)
set.seed(30012017)
#a validation set
valid<-sample(1:nrow(PERSON), 500)
PERSON$validation<- 0
PERSON[valid]$validation<-1
PERSON<-PERSON[order(validation)]
DOC<-create_matrix(c(PERSON[,paste(objectcleanpp,sep=" ")]),language="english",
                   removeNumbers=TRUE,stemWords=TRUE,removePunctuation=TRUE,removeSparseTerms=0.9999)
DOCCONT<-create_container(DOC,PERSON$label1num, trainSize=1:1200,
                          testSize=1201:nrow(PERSON), virgin=TRUE)
MOD <- train_models(DOCCONT, algorithms=c("MAXENT"))
RES <- classify_models(DOCCONT, MOD)
analytics <- create_analytics(DOCCONT, RES)
res<-data.table(analytics@document_summary)
VALID<-cbind(PERSON[validation==1],res)

#confusion matrix
table(VALID$label1,factor(VALID$MAXENTROPY_LABEL, labels=levels(VALID$label1)))
```

```
##
##             civilian security terrorist
##   civilian        96        7        13
##   security         5       93         6
##   terrorist       14        5       261
```

```
sum(diag(3) *table(VALID$MAXENTROPY_LABEL,VALID$label1))/500
```

```
## [1] 0.9
```