# A Melbourne Datathon 2017 Kaggle entry

Paul Harrison @paulfharrison

Monash Bioinformatics Platform, Monash University

14 June 2017

# Outline

Part 1:

Introduce logistic regression and decision trees

Part 2:

Kaggle competition entry

# Part 1: logistic regression and decision trees

Find a model that predicts the log odds of some outcome $y$ based on a vector predictors $x$.

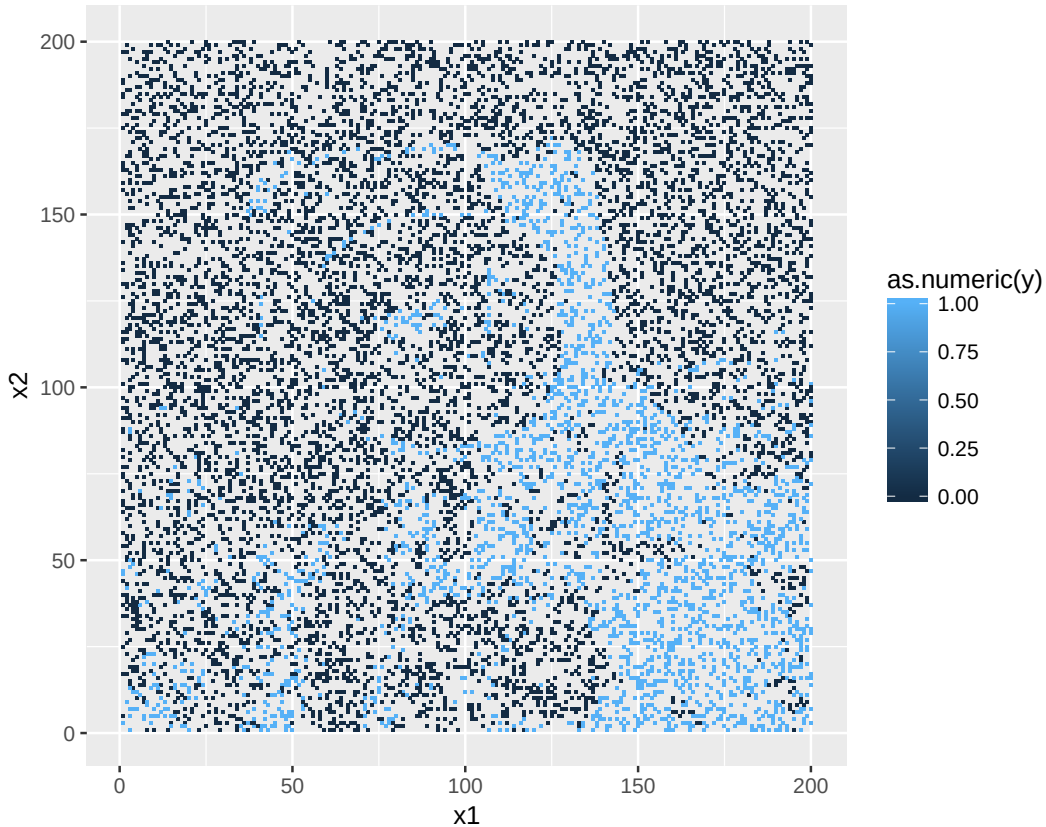
# Example dataset

```
training_xy
```

```
## # A tibble: 10,000 × 3
##       x1    x2     y
##    <dbl> <dbl> <lgl>
## 1    149   178 FALSE
## 2     92    76 FALSE
## 3    170    79  TRUE
## 4    134    76  TRUE
## 5     34    28 FALSE
## 6     10    72 FALSE
## 7    180   199 FALSE
## 8    101   154 FALSE
## 9     39    67 FALSE
## 10   166    98 FALSE
## # ... with 9,990 more rows
```

# Example dataset

```
ggplot(training_xy, aes(x=x1,y=x2,fill=as.numeric(y))) +
    geom_tile() + coord_fixed()
```



# Odds

$$\frac{p}{1-p} \quad \text{"to one"}$$

Example: p=2/3 has odds of 2 to 1.

An intuitive way to think about probability.

How much would you bet against winning \$1 on an event and still break even?

Can talk about evidence multiplicatively increasing or decreasing the odds, eg “increases the odds 2-fold”.

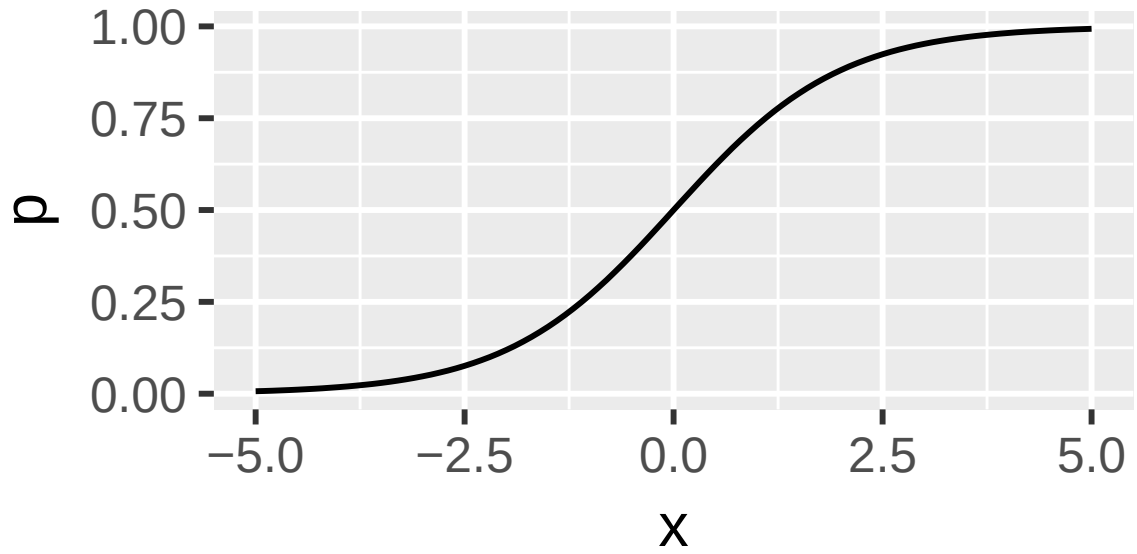
# Log odds

$$x = \log \frac{p}{1-p} \qquad p = \frac{1}{1+e^{-x}}$$

Transformation of probability which is symmetric and unbounded.

For small $p$, adding $x$s is like multiplying $p$s.

Can talk about evidence additively increasing or decreasing the log odds.



# Logistic regression

Model the log odds as a constant plus a weighted sum of predictors.

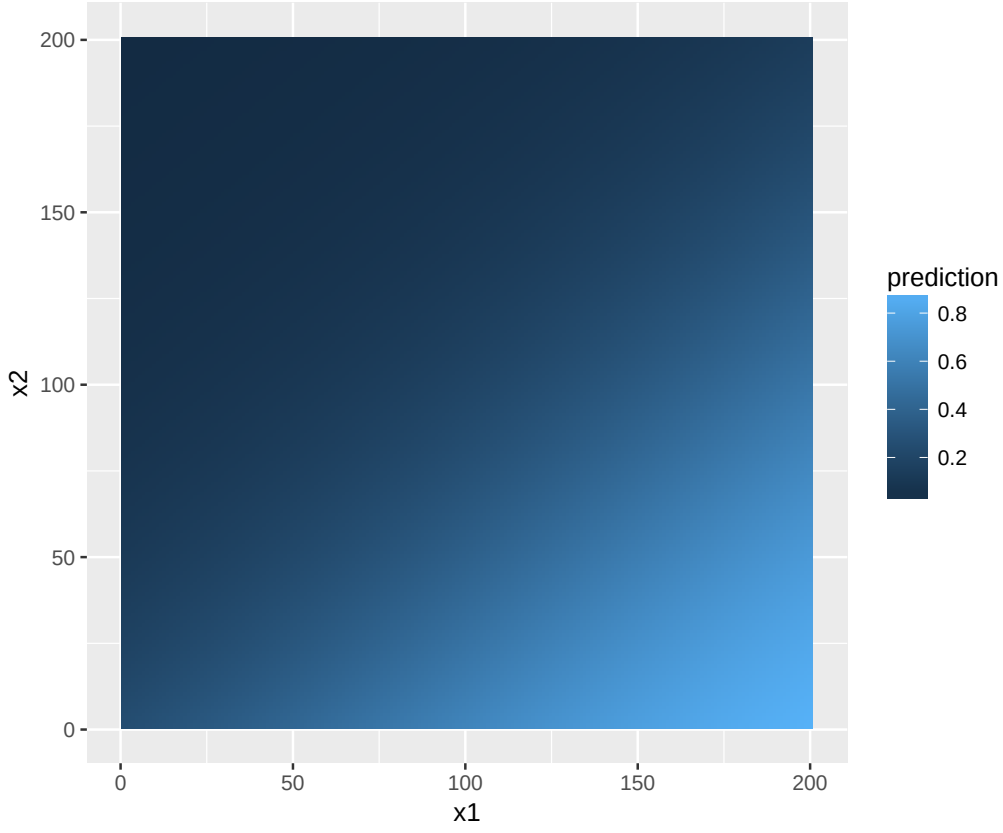
# Logistic regression

```
model <- glm(
    y ~ x1+x2, family="binomial", data=training_xy)
model
```

```
##
## Call:  glm(formula = y ~ x1 + x2, family = "binomial", d
##
## Coefficients:
## (Intercept)           x1           x2
##    -1.10181      0.01548     -0.01932
##
## Degrees of Freedom: 9999 Total (i.e. Null);  9997 Residu
## Null Deviance:       11370
## Residual Deviance: 8785  AIC: 8791
```

# Logistic regression prediction

```
prediction <- predict(model, full_x, type="response")
```



# xgboost

eXtreme Gradient Boosting of decision trees.

Model the log odds as the sum of the outputs of a stack of decision trees.

Each decision tree is a step towards better prediction of the outcome – like fitting a model by gradient descent, but each step is a decision tree.

# xgboost

```
library(xgboost)

param <- list(
    objective = "reg:logistic",
    eta = 0.1,
    max_depth = 2)

xgmodel <- xgboost(
    as.matrix(training_x), training_y,
    param=param, nrounds=100)
```

```
## [1]  train-rmse:0.477243
## [2]  train-rmse:0.457956
## [3]  train-rmse:0.441487
## [4]  train-rmse:0.427625
## [5]  train-rmse:0.415804
## [6]  train-rmse:0.405790
## [7]  train-rmse:0.396876
```

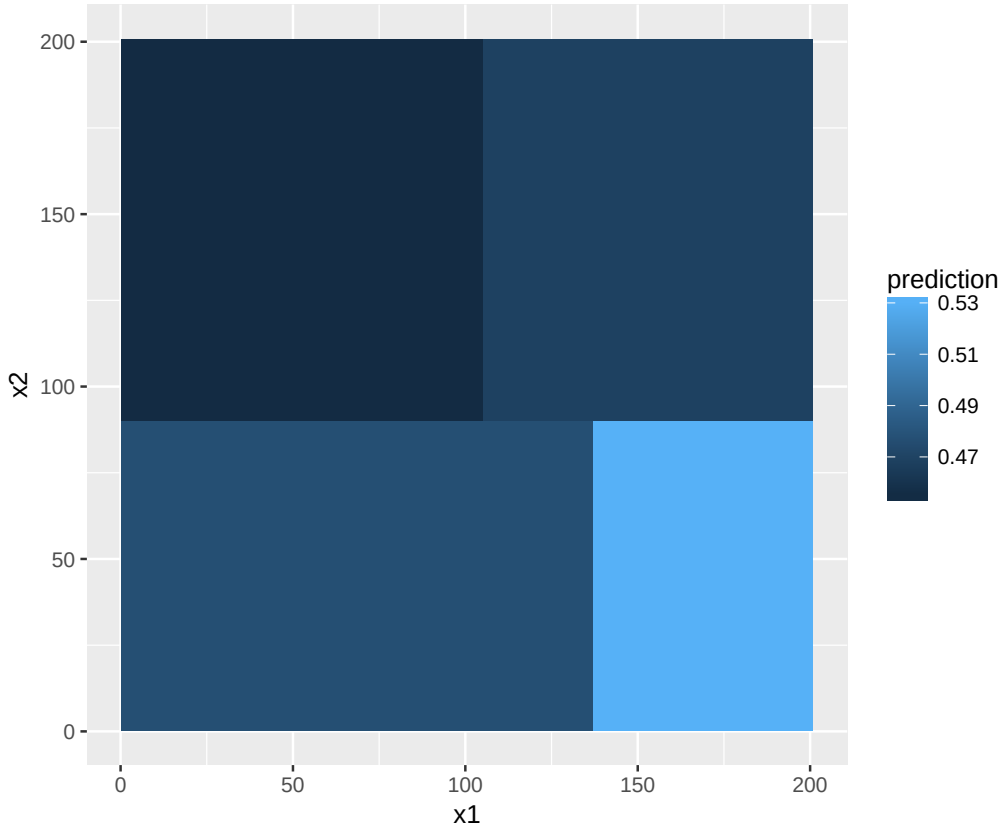
# xgboost

```
xgb.dump(xgmodel) %>% cat(sep="\n")
```

```
## booster[0]
## 0:[f1<89.5] yes=1,no=2,missing=1
## 1:[f0<137.5] yes=3,no=4,missing=3
## 3:leaf=-0.0941757
## 4:leaf=0.121394
## 2:[f0<105.5] yes=5,no=6,missing=5
## 5:leaf=-0.18299
## 6:leaf=-0.126276
## booster[1]
## 0:[f1<90.5] yes=1,no=2,missing=1
## 1:[f0<103.5] yes=3,no=4,missing=3
## 3:leaf=-0.111718
## 4:leaf=0.0709075
## 2:[f1<170.5] yes=5,no=6,missing=5
## 5:leaf=-0.126589
## 6:leaf=-0.184897
```

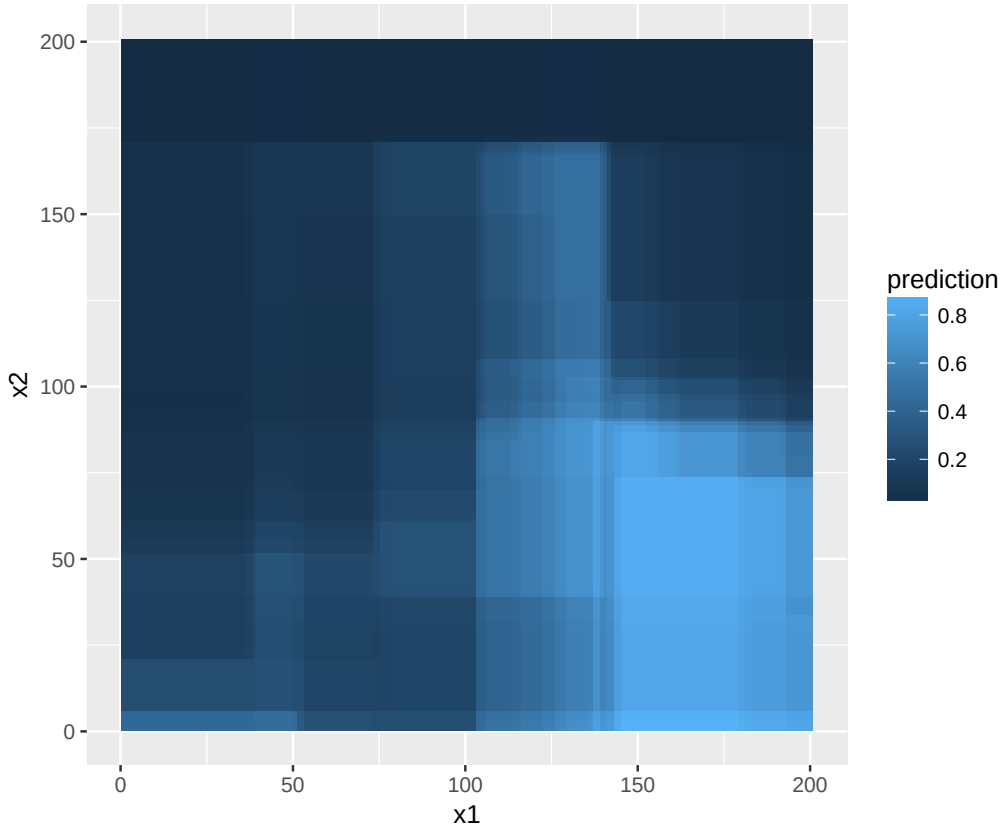
# xgboost – single tree prediction

```
prediction <- predict(
    xgmodel, as.matrix(full_x), ntreelimit=1)
```



# xgboost – many tree prediction

```
prediction <- predict(
    xgmodel, as.matrix(full_x), ntreelimit=100)
```



# xgboost – deeper trees

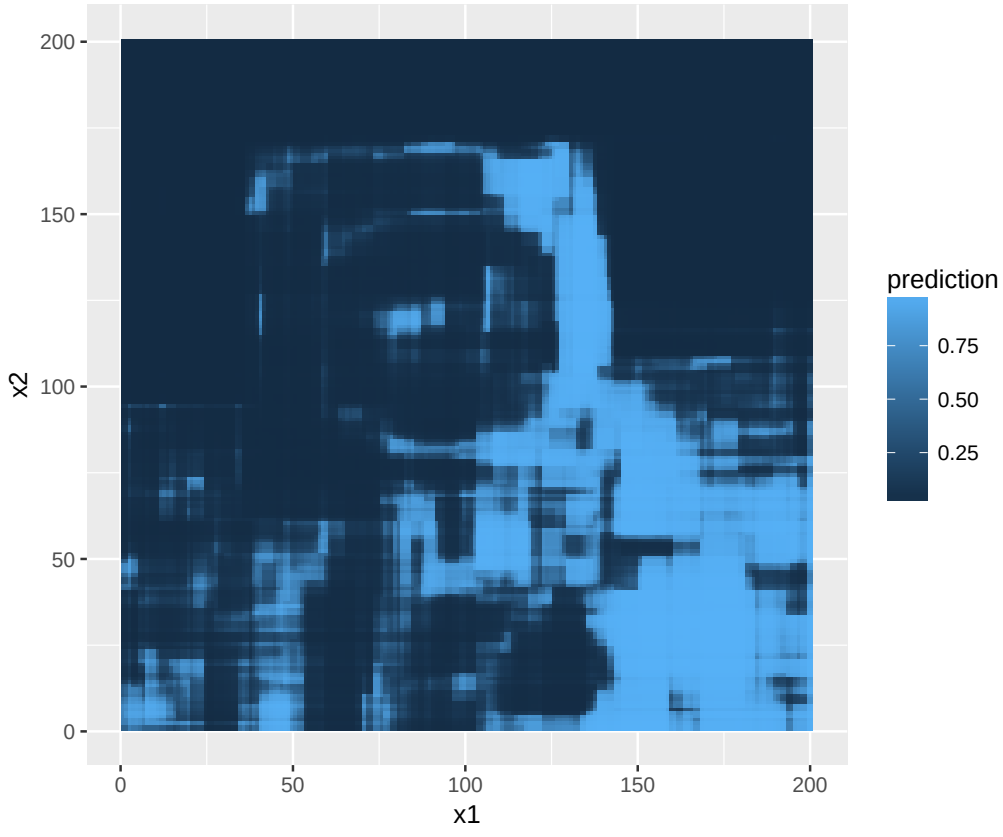
```
param <- list(
    objective = "reg:logistic",
    eta = 0.1,
    max_depth = 10)

xgmodel <- xgboost(
    as.matrix(training_x), training_y,
    param=param, nrounds=100)
```

```
## [1]  train-rmse:0.462914
## [2]  train-rmse:0.430583
## [3]  train-rmse:0.401831
## [4]  train-rmse:0.376708
## [5]  train-rmse:0.354697
## [6]  train-rmse:0.335532
## [7]  train-rmse:0.318739
## [8]  train-rmse:0.303990
## [9]  train-rmse:0.291039
```

# xgboost – deeper trees

```
prediction <- predict(
    xgmodel, as.matrix(x), ntreelimit=100)
```



# Feature engineering

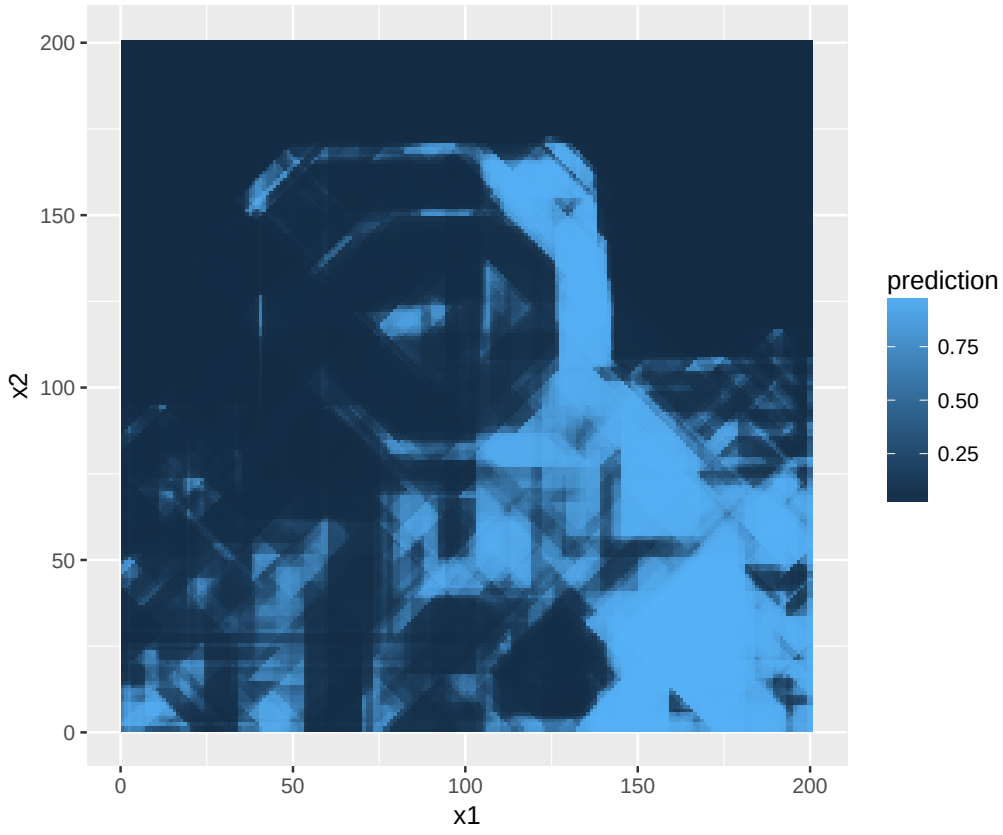
```
engineer <- function(df) {
    mutate(df,
        x3 = x1+x2,
        x4 = x1-x2
    )
}

xgmodel <- xgboost(
    as.matrix(engineer(training_x)), training_y,
    param=param, nrounds=100)
```

```
## [1]  train-rmse:0.464108
## [2]  train-rmse:0.432016
## [3]  train-rmse:0.403731
## [4]  train-rmse:0.379284
## [5]  train-rmse:0.358085
## [6]  train-rmse:0.339240
## [7]  train-rmse:0.322536
```

# Feature engineering

```
prediction <- predict(
    xgmodel, as.matrix(engineer(full_x)), ntreelimit=100)
```



# Part 2: Kaggle competition

Given 61,332,803 pharmacy drug purchases by 558,352 people in 2011-2016.

Predict purchase of diabetes-related drugs in 2016.

- Training set of 279,200 people.
- Test set of 279,152 people (2016 purchases omitted from dataset).

Scoring by ROC AUC.

# The logic

Why did you . . . ?

- Because it very slightly increased the AUC.

# The basic approach

R

Construct a large sparse matrix of predictor variables.

Apply statistical or machine learning packages:

- glmnet
- xgboost
- Stan (not used in final prediction)

Blend predictions from several packages and parameter choices.

http://tinyurl.com/medaka2017

# Bagging and blending

Bootstrap AGGregation:

- Posterior sampling of models, average over predictions.
- Each bootstrap automatically leaves out 37% of patients, a ready-made test-set for cross-validation.

Average over these test-sets can be used to blend different approaches.

# Predictor variables

- Gender, age, postcode
- How many of each drug purchased, log(1+n) transformed
  - Over all time
  - From 2015
  - ...
- Drug purchases collated by chronic illness, log(1+n) transformed
- Prescriber, 1 if ever seen else 0
- Store, proportion of purchases

7521 drugs, 11 chronic illnesses, 2665 stores, 2529 postcodes

# irlba

Augment matrix with 10-20 principal components (maximum irlba package could compute).

- Help tree methods, which otherwise always split on a single predictor.
- Unsupervised learning from test-set patients.

# glmnet

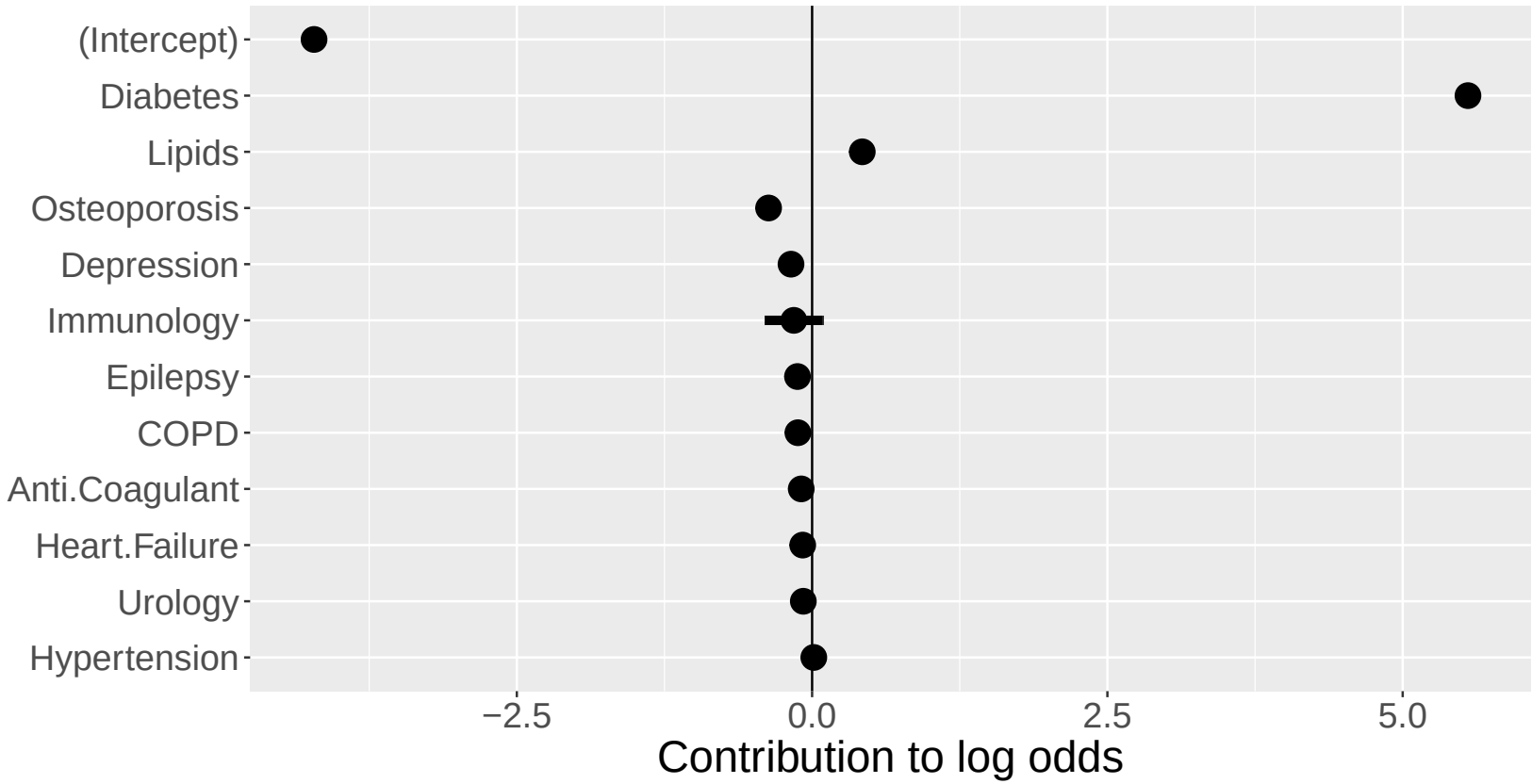
Logistic regression.

Elastic-net regularization – a mixture of L1 and L2. Mostly L2 with just a little L1 worked best.

- AUC 0.9657

# Digression: an interpretable model, illness level

Logistic regression coefficients.



# Digression: example of an interpretable model

L1 regularized logistic regression on whether each patient took each drug.

Top positive and negative coefficients, omitting diabetes drugs:

| coef | illness | MasterProductFullName | GenericIngredientName |
|---|---|---|---|
| 0.353 | Lipids | LIPIDIL TAB 145MG 30 | FENOFIBRATE |
| 0.168 | Lipids | CRESTOR TAB 10MG 30 | ROSUVASTATIN |
| 0.163 | Lipids | LIPITOR TAB 40MG 30 | ATORVASTATIN |
| 0.113 | Lipids | CRESTOR TAB 20MG 30 | ROSUVASTATIN |
| 0.109 | Lipids | LIPITOR TAB 20MG 30 | ATORVASTATIN |
| 0.103 | Lipids | VYTORIN TAB 10MG/40MG 30 | EZETIMIBE/SIMVASTATIN |
| 0.097 | Hypertension | AVAPRO HCT TAB 300MG/12.5MG 30 | IRBESARTAN/HYDROCHLOROTHIAZIDE |
| 0.095 | Lipids | LIPITOR TAB 80MG 30 | ATORVASTATIN |

| coef | illness | MasterProductFullName | GenericIngredientName |
|---|---|---|---|
| -0.109 | NA | PANTOPRAZOLE (APO) EC-TABS 40MG 30 BLISTER | PANTOPRAZOLE |
| -0.089 | NA | PARIET EC-TABS 20MG 30 | RABEPRAZOLE |
| -0.086 | NA | NEXIUM EC-TABS 20MG 30 | ESOMEPRAZOLE |
| -0.071 | NA | PANADOL OSTEO SR-TAB 665MG 96 | PARACETAMOL |
| -0.031 | COPD | SPIRIVA INH-CAP 18MCG 30 | TIOTROPIUM BROMIDE |
| -0.021 | Osteoporosis | PROTOS SACH 2G 28 | STRONTIUM RANELATE |
| -0.014 | Hypertension | MICARDIS TAB 40MG 28 | TELMISARTAN |

# xgboost

L1 and L2 regularization parameters. Learning rate and number of trees also constitute regularization.

Transformation of individual predictors not important.

Combination of predictors potentially important.

- AUC 0.9698

# Other predictions

glmnet splitting data by “diabetic prior to 2016”

- AUC 0.9664

xgboost starting from 0.5 * glmnet log odds

- AUC 0.9703

Final blended model:

- AUC 0.9705
- AUC 0.9708 on 10% of test data
- AUC 0.9706 on full test data

# Is it useful?

Baseline prediction

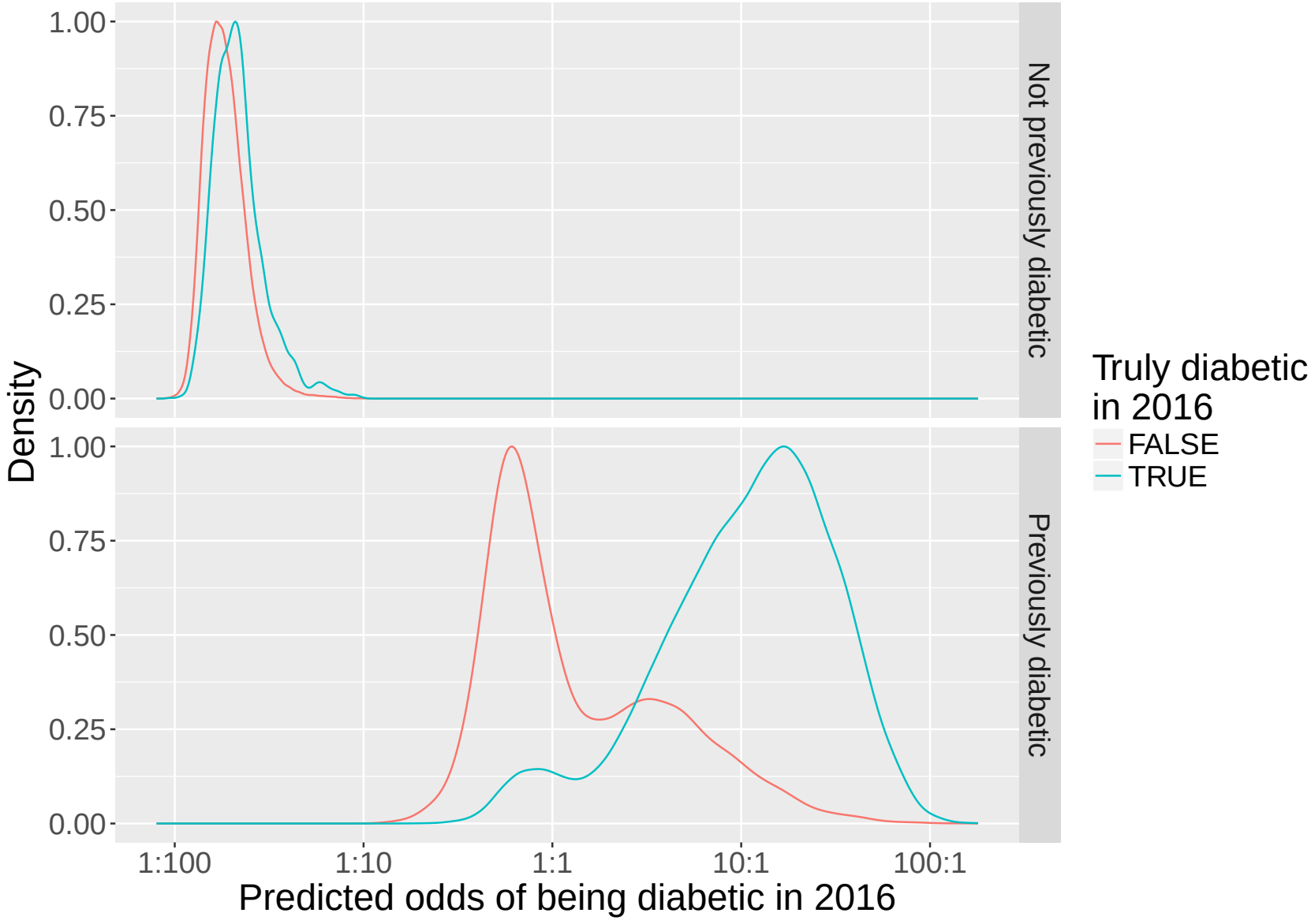
| | diabetic pre-2016 | |
|---|---|---|
| diabetic 2016 | FALSE | TRUE |
| FALSE | 213836 | 10991 |
| TRUE | 3611 | 50714 |

- AUC just from diabetic pre-2016 0.9423

My prediction, by subset

- AUC already diabetic subset 0.8907
- AUC not already diabetic subset 0.6564

# Is it useful?


Density
1.00
0.75
0.50
0.25
0.00
Not previously diabetic
Previously diabetic
1:100
1:10
1:1
10:1
100:1
Predicted odds of being diabetic in 2016
Truly diabetic in 2016
FALSE
TRUE

Thanks to my Insights Competition team-mates in team Merops:

*Di Cook*
*Earo Wang*
*Nat Tomasetti*
*Stuart Lee*

R code: http://tinyurl.com/medaka2017