Adaboost Classifier on Chromium

Decision Stump

A decision stump is a machine learning model consisting of a one-level decision tree. That is, it is a decision tree with one internal node (the root) which is immediately connected to the terminal nodes (its leaves). A decision stump makes a prediction based on the value of just a single input feature. It is also called as **weak classifier**.

Chromium's classifier uses 100 decision stump units. Each stump predicts its output given a single feature value. A decision stump takes a feature value as input and compares to a split value. If the input is bigger than the split value, it returns **1** while multiplying the associated weight to the stump. Otherwise, it returns **-1**.

AdaBoost

AdaBoost is a popular boosting technique which helps us to **combine multiple weak classifiers into a single strong classifier**. A weak classifier is simply a classifier that performs poorly, but performs better than random guessing.

AdaBoost can be applied to any classification algorithm. It's just a technique that is built on top of other classifiers, as opposed to being a classifier in general. We could even just train a bunch of weak classifiers on our own and combine the results. So, for what reason does Chromium use Adaboost?

Adaboost uses an optimally weighted majority vote and determines how much weight should be given to each decision stump classifier.

The final output is a linear combination of every **weak classifier** (decision stumps). AdaBoost evaluates the classification result by simply summing every decision stump output value. The sign of this sum identifies the predicted object class while the absolute value gives the confidence in that classification. Depending on a specified threshold, it decides whether it is confident enough in that predicted class.

Ok, but how are they trained? How do they calculate each associated weight? (bit more technical) [4]

Let's look first at the equation for the final classifier.

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

The final classifier (strong) consists of *T* weak classifiers. $\boldsymbol{h_t}(x)$ is the output of weak classifier *t*. α_t is the weight applied to classifier *t* as determined by AdaBoost.

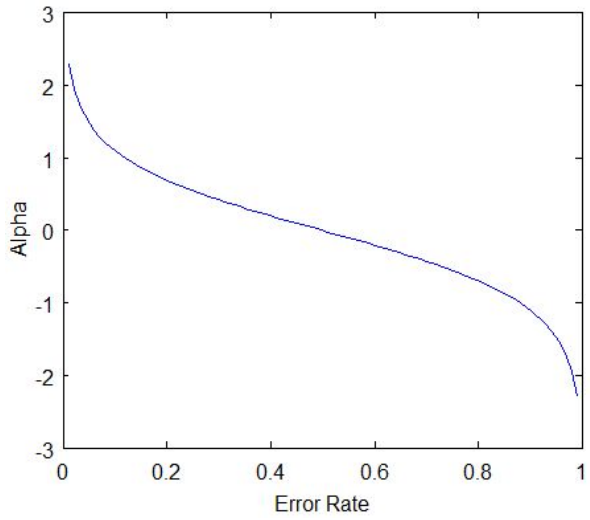
Weak classifiers are trained **one at a time**. The purpose here is to find a classifier that **reduces the minimum error after each iteration**. After each classifier is trained, we update the probabilities that each of the training samples will appear in the training set for the next classifier.

In the first step (*t = 1*), all training samples are distributed with the same probability. After the first classifier is trained, we compute the output weight (alpha) for that classifier.

$$\alpha_t = \frac{1}{2} ln\left(\frac{1 - \epsilon_t}{\epsilon_t}\right)$$

The output weight, α_t, is fairly straightforward. It's based on the classifier's error rate, ε_t which is just the number of misclassifications over the training set divided by the training set size.

Here's a plot of what α_t will look like for classifiers with different error rates.



There are three bits of intuition to take from this graph:

1. The classifier weight grows exponentially as the error approaches 0. Better classifiers are given exponentially more weight.
2. The classifier weight is zero if the error rate is 0.5. A classifier with 50% accuracy is no better than random guessing, so we ignore it.
3. The classifier weight grows exponentially negative as the error approaches 1. We give a negative weight to classifiers that have less than 50% of accuracy. “Whatever that classifier says, do the opposite!”.

After computing the alpha for the first classifier, we update the weights of every training sample using the following formula.

$$D_{t+1}(i) = \frac{D_t(i)\exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

The variable $\mathbf{D_t}$ is a vector of weights, with one weight for each training sample in the training set. Index ‘***i***’ is the training example number. This equation shows you how to update the weight for the *ith* training sample.

We can describe D_t as a distribution. This just means that each weight $D_t(i)$ represents the probability that training example *i* will be selected as part of the training set.

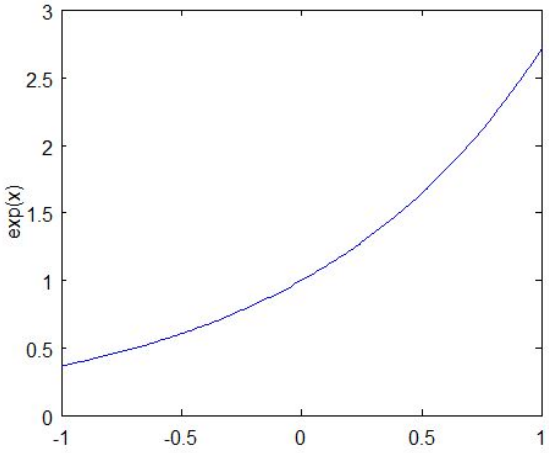
To make it a distribution, all of these probabilities should add up to 1. To ensure this, we normalize the weights by dividing each of them by the sum of all the weights, $\mathbf{Z_t}$. So, for

example, if all of the calculated weights added up to 12.2, then we would divide each of the weights by 12.2 so that they sum up to 1.0 instead.

This vector is updated for each new weak classifier that's trained. D_t refers to the weight vector used when training classifier ***'t'*** .

This equation needs to be evaluated for each of the training samples *'i'* (x_i , y_i). Each weight from the previous training round is going to be scaled up or down by this exponential term.

To understand how this exponential term behaves, let's look first at how $exp(x)$ behaves.



The function $exp(x)$ will return a fraction for negative values of *x*, and a value greater than one for positive values of x. So the weight for training sample *i* will be either increased or decreased depending on the final sign of the term "$-\alpha * y * h(x)$". For binary classifiers whose output is constrained to either -1 or +1, the terms y and $h(x)$ only contribute to the sign and not the magnitude.

y_i is the correct output for training example '*i*', and $h_t(x_i)$ is the predicted output by classifier *t* on this training example. If the predicted and actual output agree, $y * h(x)$ will always be +1 (either 1 * 1 or -1 * -1). If they disagree, $y * h(x)$ will be negative.

Ultimately, misclassifications by a classifier with a positive alpha will cause this training sample to be given a larger weight. And vice versa.

Note that by including alpha in this term, we are also incorporating the classifier's effectiveness into consideration when updating the weights. If a weak classifier misclassifies an input, we don't take that as seriously as a strong classifier's mistake.

How do weak classifiers are chosen on Chromium?

Each **decision stump** cuts the sample's space in one dimension. Each separation receives a weight that is proportional on how well it separates the samples into two distinct classes.

We do not know how they created the set of decision stumps used on Adaboost. It would even be possible to create it randomly, but they probably have an automatic and logic way of doing it. Creating and training weak classifiers is not part of the Adaboost algorithm. Adaboost only takes that in consideration in order to group a bunch of weak classifiers into one single strong classifier.

On Chromium, we noticed that some features extracted from web pages are not used at all when classifying. Some of them even have more than one weak classifier per feature, each one having a different associated weight.

Relevant code on Chromium and list of features

https://docs.google.com/document/d/1-bLtQUCEwj2WtdpPpblKT2YZV3xWHvUNwEAtMI362HU/edit?usp=sharing

Reference

[1] https://en.wikipedia.org/wiki/Decision_stump

[2] https://en.wikipedia.org/wiki/AdaBoost

[3] http://www.cs.utexas.edu/~grauman/courses/spring2007/395T/papers/viola_cvpr2001.pdf

[4] http://mccormickml.com/2013/12/13/adaboost-tutorial/

[5] http://machinelearningmastery.com/boosting-and-adaboost-for-machine-learning/

Appendix

Features

0: 'opengraph', opengraph, (bool)
Whether the page is opengraph

1: 'forum', 'forum' in path, (bool)
Whether the page's path has the word 'forum'

2: 'index', 'index' in path, (bool)
Whether the page's path has the word index

3: 'search', 'search' in path, (bool)
Whether the page's path has the word search'

4: 'view', 'view' in path, (bool)
Whether the page's path has the word 'view'

5: 'archive', 'archive' in path, (bool)
Whether the page's path has the word 'archive

6: 'asp', '.asp' in path, (bool)
Whether the page's path has the word .asp'

7: 'phpbb', 'phpbb' in path, (bool)
Whether the page's path has the word 'phpbb'

8: 'php', path.endswith('.php'), (bool)
Whether the page's path ends with the word '.php'

9: 'pathLength', len(path), (int)
The length of the path

10: 'domain', len(path) < 2, (bool)
Whether the page's path is less than 2 chars

11: 'pathComponents', CountMatches(path, r'\/.'), (int)
How many components in the path

12: 'slugDetector', CountMatches(path, r'[^\w/]'), (int)

?

13: 'pathNumbers', CountMatches(path, r'\d+'), (int)
How many numbers are in the path

14: 'lastSegmentLength', len(GetLastSegment(path)), (int)
The length of the last segment

15: 'formCount', numForms, (int)
How many forms

16: 'anchorCount', numAnchors, (int)
How many achors

17: 'elementcount', numElements, (int)
How many elements

18: 'anchorratio', float(numAnchors) / max(1, numElements), (float)
Ratio between #anchors and the #elements

19: 'mozScore' (float)
MozScore

20: 'mozScoreAllSqrt' (float)
Squared MozScore

21: 'mozScoreAllLinear' (int)
Linear MozScore

Statistics

In this list each entry contains statistics about the corresponding feature.


```
[
 // 'opengraph', opengraph, (bool)
 { min: 0,
   max: 1,
   sum: 660,
   avg: 0.11520335137022168,
   vrn: 0.10194933451390346,
   std: 0.3192950587057424,
```

```
  rng: 1 },

// 'forum', 'forum' in path, (bool)
{ min: 0,
  max: 1,
  sum: 4,
  avg: 0.0006982021295164951,
  vrn: 0.0006978364510268369,
  std: 0.02641659423595019,
  rng: 1 },

// 'index', 'index' in path, (bool)
{ min: 0,
  max: 1,
  sum: 138,
  avg: 0.02408797346831908,
  vrn: 0.023511847007922432,
  std: 0.15333573297807146,
  rng: 1 },

// 'search', 'search' in path, (bool)
{ min: 0,
  max: 1,
  sum: 1339,
  avg: 0.2337231628556467,
  vrn: 0.17912791287295596,
  std: 0.4232350562901849,
  rng: 1 },


// 'view', 'view' in path, (bool)
{ min: 0,
  max: 1,
  sum: 54,
  avg: 0.009425728748472683,
  vrn: 0.009338514428697462,
  std: 0.09663598930366192,
  rng: 1 },

// 'archive', 'archive' in path, (bool)
{ min: 0,
  max: 1,
  sum: 6,
  avg: 0.0010473031942747426,
  vrn: 0.001046388998050631,
  std: 0.0323479365346637,
  rng: 1 },
```

```
// 'asp', '.asp' in path, (bool)
{ min: 0,
  max: 1,
  sum: 315,
  avg: 0.054983417699423984,
  vrn: 0.051969312748726765,
  std: 0.2279677888402806,
  rng: 1 },

// 'phpbb', 'phpbb' in path, (bool)
{ min: 0,
  max: 0,
  sum: 0,
  avg: 0,
  vrn: 0,
  std: 0,
  rng: 0 },

// 'php', path.endswith('.php'), (bool)
{ min: 0,
  max: 1,
  sum: 255,
  avg: 0.04451038575667656,
  vrn: 0.04253663610894335,
  std: 0.20624411775598195,
  rng: 1 },

// 'pathLength', len(path), (int)
{ min: 1,
  max: 336,
  sum: 146490,
  avg: 25.56990748821784,
  vrn: 733.9211333887735,
  std: 27.090978819318682,
  rng: 335 },

// 'domain', len(path) < 2, (bool)
{ min: 0,
  max: 1,
  sum: 609,
  avg: 0.10630127421888637,
  vrn: 0.09501789874313765,
  std: 0.3082497343764268,
  rng: 1 },

// 'pathComponents', CountMatches(path, r'\/.'), (int)
{ min: 0,
  max: 14,
```

```
  sum: 10892,
  avg: 1.901204398673416,
  vrn: 2.2325561609024547,
  std: 1.4941740731596351,
  rng: 14 },

// 'slugDetector', CountMatches(path, r'[^\w/]'), (int)
{ min: 0,
  max: 109,
  sum: 10349,
  avg: 1.8064234595915518,
  vrn: 14.373659849282241,
  std: 3.7912609840635136,
  rng: 109 },

// 'pathNumbers', CountMatches(path, r'\d+'), (int)
{ min: 0,
  max: 72,
  sum: 4658,
  avg: 0.8130563798219584,
  vrn: 7.3129859257667436,
  std: 2.7042533028114697,
  rng: 72 },

// 'lastSegmentLength', len(GetLastSegment(path)),
{ min: 0,
  max: 330,
  sum: 77591,
  avg: 13.543550357828591,
  vrn: 331.89269975310657,
  std: 18.217922487295485,
  rng: 330 },

// 'formCount', numForms, (int)
{ min: 0,
  max: 90,
  sum: 9462,
  avg: 1.651597137371269,
  vrn: 5.687253471751841,
  std: 2.384796316617384,
  rng: 90 },

// 'anchorCount', numAnchors, (int)
{ min: 0,
  max: 13546,
  sum: 793050,
  avg: 138.42729970326408,
  vrn: 77034.22206185818,
```

```
  std: 277.5503955354021,
  rng: 13546 },

// 'elementcount', numElements, (int)
{ min: 0,
  max: 27821,
  sum: 4486882,
  avg: 783.1876418223076,
  vrn: 941192.7771098216,
  std: 970.1509042977909,
  rng: 27821 },

// 'anchorratio', float(numAnchors) / max(1, numElements), (float)
{ min: 0,
  max: 0.612,
  sum: 826.8879991022063,
  avg: 0.1443337404611985,
  vrn: 0.008172160011498989,
  std: 0.0904000000636006,
  rng: 0.612 },

// 'mozScore' (float)
{ min: 0,
  max: 6033.798624065624,
  sum: 213890.64730361325,
  avg: 37.334726357761085,
  vrn: 28300.311533070482,
  std: 168.2269643460004,
  rng: 6033.798624065624 },

// 'mozScoreAllSqrt' (float)
{ min: 0,
  max: 7322.83106016992,
  sum: 448878.3359420426,
  avg: 78.3519525121387,
  vrn: 59279.60970693959,
  std: 243.47404318928864,
  rng: 7322.83106016992 },

// 'mozScoreAllLinear' (int)
{ min: 0,
  max: 270255,
  sum: 7976324,
  avg: 1392.271600628382,
  vrn: 41458281.133274384,
  std: 6438.810537146934,
  rng: 270255
}
```

]

Class distribution

```
{ '1': 1689, '-1': 4040 }
```

Covariance and Correlation

To keep in mind:

- If *cor* = 0 then the points are a complete jumble with absolutely no straight line relationship between the data.
- If *cor* = -1 or *cor* = 1 then all of the data points line up perfectly on a line.
- If *cor* is a value other than these extremes, then the result is a less than perfect fit of a straight line. In real world data sets this is the most common result.
- If *cor* is positive then the line is going up with positive slope. If *r* is negative then the line is going down with negative slope.

Also:

- Correlation does not completely tell us everything about the data. Means and standard deviations continue to be important.
- The data may be described by a curve more complicated than a straight line, but this will not show up in the calculation of *r*.
- Outliers strongly influence the correlation coefficient. If we see any outliers in our data, we should be careful about what conclusions we draw from the value of *r* .
- Just because two sets of data are correlated, it doesn't mean that one is the cause of the other.

```
var 0 with 1: { cor: -0.009537906754381013, cov: -0.00008044926771662415 }
var 0 with 2: { cor: 0.036021856951522366, cov: 0.001763606409026834 }
var 0 with 3: { cor: -0.19540670792805673, cov: -0.02640664935138492 }
var 0 with 4: { cor: 0.01572379655972535, cov: 0.0004851639361047258 4 }
var 0 with 5: { cor: 0.07283037062409391, cov: 0.0007522311263579725 }
var 0 with 6: { cor: -0.08224070622719043, cov: -0.0059862178215114486 }
```

var 0 with 7: { cor: NaN, cov: 0 }
var 0 with 8: { cor: -0.05402072004649084, cov: -0.003557411766656171 }
var 0 with 9: { cor: 0.3847407425399939, cov: 3.328013452824229 }
var 0 with 10: { cor: -0.10848291054804744, cov: -0.010677171959577106 }
var 0 with 11: { cor: 0.21890400423170583, cov: 0.10443524735956214 }
var 0 with 12: { cor: 0.34926277951482054, cov: 0.42279338629007385 }
var 0 with 13: { cor: 0.09065804691162574, cov: 0.07827911824677553 }
var 0 with 14: { cor: 0.2823077318196407, cov: 1.6421537646356132 }
var 0 with 15: { cor: 0.18959662121771778, cov: 0.14436904492579264 }
var 0 with 16: { cor: 0.06577776758618431, cov: 5.829256668269166 }
var 0 with 17: { cor: 0.11924732470714325, cov: 36.93857479002749 }
var 0 with 18: { cor: 0.15692495561114514, cov: 0.004529524810635528 }
var 0 with 19: { cor: 0.10748534441301587, cov: 5.773471923337218 }
var 0 with 20: { cor: 0.14750168604262542, cov: 11.466789762786664 }
var 0 with 21: { cor: 0.08961738429307316, cov: 184.24262283262829 }
var 1 with 2: { cor: -0.004152762051327052, cov: -0.00001682121052256765 }
var 1 with 3: { cor: -0.01459823257882847, cov: -0.00016321449920085455 }
var 1 with 4: { cor: -0.00257843505595631, cov: -0.000006582212813179315 }
var 1 with 5: { cor: -0.0008558664500921961, cov: -7.313569792421214e-7 }
var 1 with 6: { cor: -0.0063758545534219405, cov: -0.00003839624141021124 }
var 1 with 7: { cor: NaN, cov: 0 }
var 1 with 8: { cor: -0.00570505642156376, cov: -0.00003108267161778964 }
var 1 with 9: { cor: 0.029693318218765027, cov: 0.02125006460320109 }
var 1 with 10: { cor: -0.009116243480540664, cov: -0.00007423273339307188 }
var 1 with 11: { cor: 0.010593941288842439, cov: 0.0004181533528816949 }
var 1 with 12: { cor: 0.029240247825626883, cov: 0.0029284752377158265 }
var 1 with 13: { cor: 0.011602801923821203, cov: 0.0008288712431410651 }
var 1 with 14: { cor: 0.0318598276973874, cov: 0.015332716230566283 }
var 1 with 15: { cor: 0.006633191366715045, cov: 0.0004178790940144839 }
var 1 with 16: { cor: 0.004388155134823185, cov: 0.032173673391575355 }
var 1 with 17: { cor: 0.00648000413881777, cov: 0.16607008252632174 }
var 1 with 18: { cor: -0.0026058050844834724, cov:
-0.0000062228192043377744 }
var 1 with 19: { cor: 0.011508903709238705, cov: 0.05114537768831029 }
var 1 with 20: { cor: 0.008211068292215636, cov: 0.0528115795923896 }
var 1 with 21: { cor: 0.010009315159971735, cov: 1.7024988822427083 }
var 2 with 3: { cor: -0.08138640499819152, cov: -0.005281738211256104 }
var 2 with 4: { cor: -0.015325291504691847, cov: -0.00022708634205467204 }
var 2 with 5: { cor: -0.005086962654518144, cov: -0.00002523181578385439 }
var 2 with 6: { cor: 0.017042207729518744, cov: 0.0005957207328001587 }
var 2 with 7: { cor: NaN, cov: 0 }
var 2 with 8: { cor: 0.17034669952119735, cov: 0.0053871450358902095 }
var 2 with 9: { cor: 0.015522807681029616, cov: 0.06448197741374563 }

var 2 with 10: { cor: -0.054183675654090636, cov: -0.002561029302061578 }
var 2 with 11: { cor: 0.11402028064583307, cov: 0.026123218048721572 }
var 2 with 12: { cor: -0.0012872604424116869, cov: -0.0007483305558016118 }
var 2 with 13: { cor: 0.004125279858760159, cov: 0.0017105830280321234 }
var 2 with 14: { cor: -0.028248979376827188, cov: -0.07891235149796655 }
var 2 with 15: { cor: 0.04730276721887463, cov: 0.017297415335678416 }
var 2 with 16: { cor: -0.007236062529917209, cov: -0.3079551953649485 }
var 2 with 17: { cor: 0.030857178099055693, cov: 4.59027678570546 }
var 2 with 18: { cor: -0.02875018417481993, cov: -0.00039852212323890986 }
var 2 with 19: { cor: -0.004765517879354207, cov: -0.12292751007949922 }
var 2 with 20: { cor: -0.005248632357525272, cov: -0.19594861351924486 }
var 2 with 21: { cor: -0.005801947610483043, cov: -5.728261327988385 }
var 3 with 4: { cor: -0.0538731010731384, cov: -0.002203395739211623 }
var 3 with 5: { cor: -0.01788223428952224, cov: -0.00024482174880129955 }
var 3 with 6: { cor: -0.1223587401017308, cov: -0.01180565577854919 }
var 3 with 7: { cor: NaN, cov: 0 }
var 3 with 8: { cor: -0.1051997314009048, cov: -0.009182857284949315 }
var 3 with 9: { cor: -0.3637700322650698, cov: -4.170933332179437 }
var 3 with 10: { cor: -0.1904722421842973, cov: -0.02484940750333345 }
var 3 with 11: { cor: -0.3171214220632365, cov: -0.20054341651949684 }
var 3 with 12: { cor: -0.25283116910095527, cov: -0.4056915175267365 }
var 3 with 13: { cor: -0.15584179452310282, cov: -0.17836635694510394 }
var 3 with 14: { cor: -0.2188542266797204, cov: -1.6874675155608463 }
var 3 with 15: { cor: -0.00942474159401066, cov: -0.009512668809378904 }
var 3 with 16: { cor: 0.06049650554602834, cov: 7.106467475092482 }
var 3 with 17: { cor: 0.20866788714075218, cov: 85.67942520948539 }
var 3 with 18: { cor: 0.006696077656479347, cov: 0.00025619493844950415 }
var 3 with 19: { cor: -0.1204790097816064, cov: -8.578051127225768 }
var 3 with 20: { cor: -0.1489969129013985, cov: -15.353647690309234 }
var 3 with 21: { cor: -0.11279428670684728, cov: -307.37913289828884 }
var 4 with 5: { cor: -0.0031584768582052464, cov: -0.000099873319219769577
}
var 4 with 6: { cor: -0.015604626700210934, cov: -0.0003437682534512844 }
var 4 with 7: { cor: NaN, cov: 0 }
var 4 with 8: { cor: 0.013983918927305015, cov: 0.00027870795550621717 }
var 4 with 9: { cor: 0.029490129740737413, cov: 0.07720408443370368 }
var 4 with 10: { cor: -0.033642449781677894, cov: -0.0010021419008064783 }
var 4 with 11: { cor: 0.06327742743054153, cov: 0.00913669037563495 }
var 4 with 12: { cor: -0.005978740449756133, cov: -0.0021904446260377786 }
var 4 with 13: { cor: 0.012088383906060153, cov: 0.0031590355254212045 }
var 4 with 14: { cor: 0.00908831906982659, cov: 0.016000049000916644 }
var 4 with 15: { cor: -0.025897521806138406, cov: -0.00596826910231307 }
var 4 with 16: { cor: -0.005910681695393827, cov: -0.15853250418580078 }

var 4 with 17: { cor: -0.022560409311188422, cov: -2.115072042319248 }
var 4 with 18: { cor: -0.03778698314079906, cov: -0.0003301030581067597 }
var 4 with 19: { cor: 0.0018846453001631635, cov: 0.03063826237773166 }
var 4 with 20: { cor: 0.0276042706481588, cov: 0.6494830802468228 }
var 4 with 21: { cor: 0.006254160396101582, cov: 3.891468848824859 }
var 5 with 6: { cor: -0.007810159504209719, cov: -0.000057594362115317384 }
var 5 with 7: { cor: NaN, cov: 0 }
var 5 with 8: { cor: -0.006988459391536333, cov: -0.00004662400742668736 }
var 5 with 9: { cor: 0.012467090798648373, cov: 0.010925376234408733 }
var 5 with 10: { cor: -0.011167023191271355, cov: -0.00011134910008961087 }
var 5 with 11: { cor: 0.04909726450210562, cov: 0.002373040085188636 }
var 5 with 12: { cor: 0.00022984212976049487, cov: 0.0000281877169082775 }
var 5 with 13: { cor: 0.024191633787269955, cov: 0.002116211892644708 }
var 5 with 14: { cor: -0.0021511270762748043, cov: -0.00126768543068632 }
var 5 with 15: { cor: 0.018309227976788515, cov: 0.001412433166161282 }
var 5 with 16: { cor: 0.021495237148510284, cov: 0.19298816371864905 }
var 5 with 17: { cor: 0.024398733372581444, cov: 0.7656903193202084 }
var 5 with 18: { cor: 0.025707040660895953, cov: 0.00007517390272214674 }
var 5 with 19: { cor: 0.002890082037218503, cov: 0.01572723445972056 }
var 5 with 20: { cor: 0.022613352482220343, cov: 0.17810011605685747 }
var 5 with 21: { cor: 0.006337898428210358, cov: 1.3200716473864482 }
var 6 with 7: { cor: NaN, cov: 0 }
var 6 with 8: { cor: -0.052061160509484665, cov: -0.0024477603899011554 }
var 6 with 9: { cor: 0.028338323390723563, cov: 0.17501381655226111 }
var 6 with 10: { cor: -0.08318974958601297, cov: -0.00584582775470486 }
var 6 with 11: { cor: 0.03850183592916903, cov: 0.013114632405354822 }
var 6 with 12: { cor: 0.00464115186428638, cov: 0.004011279718690149 }
var 6 with 13: { cor: -0.035996914996545205, cov: -0.022191473401521992 }
var 6 with 14: { cor: 0.05720224767983175, cov: 0.2375666266208044 }
var 6 with 15: { cor: -0.06751722992054579, cov: -0.03670619732401385 }
var 6 with 16: { cor: -0.05822056903170124, cov: -3.6837638628715283 }
var 6 with 17: { cor: -0.10196464067815882, cov: -22.550821783175053 }
var 6 with 18: { cor: -0.05970596810839138, cov: -0.001230437793599212 }
var 6 with 19: { cor: -0.018214560408434173, cov: -0.6985343858069883 }
var 6 with 20: { cor: -0.021376143873415437, cov: -1.186466604131607 }
var 6 with 21: { cor: -0.012438046768048463, cov: -18.257079992657125 }
var 7 with 8: { cor: NaN, cov: 0 }
var 7 with 9: { cor: NaN, cov: 0 }
var 7 with 10: { cor: NaN, cov: 0 }
var 7 with 11: { cor: NaN, cov: 0 }
var 7 with 12: { cor: NaN, cov: 0 }
var 7 with 13: { cor: NaN, cov: 0 }
var 7 with 14: { cor: NaN, cov: 0 }

var 7 with 15: { cor: NaN, cov: 0 }
var 7 with 16: { cor: NaN, cov: 0 }
var 7 with 17: { cor: NaN, cov: 0 }
var 7 with 18: { cor: NaN, cov: 0 }
var 7 with 19: { cor: NaN, cov: 0 }
var 7 with 20: { cor: NaN, cov: 0 }
var 7 with 21: { cor: NaN, cov: 0 }
var 8 with 9: { cor: 0.0006459614974756691, cov: 0.0036092162193528746 }
var 8 with 10: { cor: -0.0744374281294088, cov: -0.0047323367538080864 }
var 8 with 11: { cor: 0.04996291174177676, cov: 0.015396801385873068 }
var 8 with 12: { cor: -0.026265114883999505, cov: -0.020537357226928184 }
var 8 with 13: { cor: -0.047368681312464585, cov: -0.02641923478606829 }
var 8 with 14: { cor: -0.005092702283778527, cov: -0.019135010692440054 }
var 8 with 15: { cor: -0.07246511357474407, cov: -0.0356419814995937 }
var 8 with 16: { cor: -0.07899010413714218, cov: -4.521641310113845 }
var 8 with 17: { cor: -0.11780161040962228, cov: -23.570678886991768 }
var 8 with 18: { cor: -0.12310409870050217, cov: -0.002295210460682975 }
var 8 with 19: { cor: -0.025426719575597806, cov: -0.8822009324801171 }
var 8 with 20: { cor: -0.028474473620279734, cov: -1.4298482337351397 }
var 8 with 21: { cor: -0.0215351656709137, cov: -28.597985014008483 }
var 9 with 10: { cor: -0.3128170361007821, cov: -2.6122684462857553 }
var 9 with 11: { cor: 0.7332673402015272, cov: 29.681663344680743 }
var 9 with 12: { cor: 0.7881958849518234, cov: 80.95478830384552 }
var 9 with 13: { cor: 0.5977943300776944, cov: 43.79493207400111 }
var 9 with 14: { cor: 0.7581961864164972, cov: 374.20117110365436 }
var 9 with 15: { cor: 0.09815425096906472, cov: 6.3413993272490945 }
var 9 with 16: { cor: -0.010447077787761938, cov: -78.55274677569061 }
var 9 with 17: { cor: -0.05858783877711139, cov: -1539.8253579870495 }
var 9 with 18: { cor: 0.02285473233490315, cov: 0.0559717991317665 }
var 9 with 19: { cor: 0.10256722874981526, cov: 467.44328614497476 }
var 9 with 20: { cor: 0.11822022484009757, cov: 779.7747094236338 }
var 9 with 21: { cor: 0.09284040653732303, cov: 16194.493754181092 }
var 10 with 11: { cor: -0.4388733265048609, cov: -0.20213573302935747 }
var 10 with 12: { cor: -0.1643415790241343, cov: -0.19205863947123336 }
var 10 with 13: { cor: -0.1037014587204176, cov: -0.08644401803623818 }
var 10 with 14: { cor: -0.2564160756894292, cov: -1.439948004175547 }
var 10 with 15: { cor: 0.008116733535293056, cov: 0.005966714968731681 }
var 10 with 16: { cor: 0.018950354053467785, cov: 1.6212944274986103 }
var 10 with 17: { cor: 0.03247828848806514, cov: 9.712591852341971 }
var 10 with 18: { cor: 0.01270688756046972, cov: 0.0003540872825091554 }
var 10 with 19: { cor: -0.03872292645163698, cov: -2.0080128629622025 }
var 10 with 20: { cor: -0.017427678977948894, cov: -1.307961408738611 }
var 10 with 21: { cor: -0.028292328057190858, cov: -56.15352737127695 }

var 11 with 12: { cor: 0.36888057663850593, cov: 2.0896361169917057 }
var 11 with 13: { cor: 0.37594738924706983, cov: 1.5190624844586467 }
var 11 with 14: { cor: 0.35453112506506623, cov: 9.650602217620602 }
var 11 with 15: { cor: 0.02914193383637492, cov: 0.10384147691203255 }
var 11 with 16: { cor: -0.03403991584359614, cov: -14.116646013958107 }
var 11 with 17: { cor: -0.08089747192355858, cov: -117.26689852105005 }
var 11 with 18: { cor: -0.0628357576638999, cov: -0.0084874354271455 }
var 11 with 19: { cor: 0.069386511200488, cov: 17.44101902651441 }
var 11 with 20: { cor: 0.07840123509843021, cov: 28.521789380822156 }
var 11 with 21: { cor: 0.06825497715920674, cov: 656.660915844232 }
var 12 with 13: { cor: 0.6942638338637821, cov: 7.117960810967429 }
var 12 with 14: { cor: 0.7278523727285573, cov: 50.27196182731138 }
var 12 with 15: { cor: 0.11046004353099412, cov: 0.9987118061006323 }
var 12 with 16: { cor: 0.018509681387594328, cov: 19.477108130398108 }
var 12 with 17: { cor: -0.004903468252685054, cov: -18.035423397182345 }
var 12 with 18: { cor: 0.09019265327080812, cov: 0.030911727442236248 }
var 12 with 19: { cor: 0.10686681165592628, cov: 68.15883242017541 }
var 12 with 20: { cor: 0.11630855902849271, cov: 107.3613650125497 }
var 12 with 21: { cor: 0.09594874478361069, cov: 2342.225070722161 }
var 13 with 14: { cor: 0.522235233767771, cov: 25.728376821444563 }
var 13 with 15: { cor: -0.011021520993187707, cov: -0.0710788173665121 }
var 13 with 16: { cor: -0.014971316360085581, cov: -11.23696962601327 }
var 13 with 17: { cor: -0.05125182710765861, cov: -134.46090007128296 }
var 13 with 18: { cor: -0.0439112013781056, cov: -0.010734729834239792 }
var 13 with 19: { cor: 0.03648819385747191, cov: 16.599512875710918 }
var 13 with 20: { cor: 0.038439305811309124, cov: 25.3090341958641 }
var 13 with 21: { cor: 0.04029503037621009, cov: 701.624106891222 }
var 14 with 15: { cor: 0.06019847953755633, cov: 2.6153852154725232 }
var 14 with 16: { cor: 0.013223569476075425, cov: 66.86354551746442 }
var 14 with 17: { cor: -0.025250789939907346, cov: -446.2858443845436 }
var 14 with 18: { cor: 0.028241301203368367, cov: 0.046510604430927374 }
var 14 with 19: { cor: 0.08344580675530294, cov: 255.7401855079297 }
var 14 with 20: { cor: 0.07512484100341038, cov: 333.2230871487472 }
var 14 with 21: { cor: 0.06557427244491465, cov: 7691.976996446187 }
var 15 with 16: { cor: 0.1776697913613322, cov: 117.599841167548 }
var 15 with 17: { cor: 0.3208613346014208, cov: 742.3487313333275 }
var 15 with 18: { cor: 0.15601521412777847, cov: 0.03363463154579668 }
var 15 with 19: { cor: 0.027319316001873963, cov: 10.960155656247833 }
var 15 with 20: { cor: 0.10262459900504219, cov: 59.587536810514976 }
var 15 with 21: { cor: 0.03527654250882837, cov: 541.6801876491238 }
var 16 with 17: { cor: 0.833266017070066, cov: 224370.0133821251 }
var 16 with 18: { cor: 0.4066985642035143, cov: 10.204293008375439 }
var 16 with 19: { cor: 0.3045759482673429, cov: 14221.095855932732 }

```
var 16 with 20: { cor: 0.2782198492286145, cov: 18801.07272432461 }
var 16 with 21: { cor: 0.2510826556128209, cov: 448708.4106357642 }
var 17 with 18: { cor: 0.2895194368329931, cov: 25.391329946224495 }
var 17 with 19: { cor: 0.218094635221586, cov: 35594.253058676906 }
var 17 with 20: { cor: 0.23570220469388733, cov: 55674.40770307219 }
var 17 with 21: { cor: 0.18569333423158157, cov: 1159955.2990623827 }
var 18 with 19: { cor: 0.22682988495344958, cov: 3.449564830794818 }
var 18 with 20: { cor: 0.2117567798114261, cov: 4.6607780568293125 }
var 18 with 21: { cor: 0.20049341825966702, cov: 116.70089780645772 }
var 19 with 20: { cor: 0.8966946588253599, cov: 36727.62612856614 }
var 19 with 21: { cor: 0.8400425219163845, cov: 909918.5615124725 }
var 20 with 21: { cor: 0.7922938132618347, cov: 1242065.7280934385 }
```

Covariance and Correlation between the features and the class:

```
var 0 with class: { cor: 0.411782354335264, cov: 0.1199097554244685 }
var 1 with class: { cor: 0.011894861134705925, cov: 0.0002865700430329667 }
var 2 with class: { cor: 0.010774853455078927, cov: 0.00150677821648436 }
var 3 with class: { cor: -0.3516661581981669, cov: -0.13573967250810526 }
var 4 with class: { cor: 0.02012576693748, cov: 0.0017737235139071784 }
var 5 with class: { cor: 0.03824167227399942, cov: 0.0011281790868958716 }
var 6 with class: { cor: 0.020376396297044554, cov: 0.004236385302260207 }
var 7 with class: { cor: NaN, cov: 0 }
var 8 with class: { cor: -0.03188779408026477, cov: -0.0059979195331795404
}
var 9 with class: { cor: 0.40620590995132644, cov: 10.036112518296145 }
var 10 with class: { cor: -0.12487654919619921, cov: -0.035105744467773335
}
var 11 with class: { cor: 0.2907874774732249, cov: 0.39625201488850315 }
var 12 with class: { cor: 0.3444483984112482, cov: 1.190974433222689 }
var 13 with class: { cor: 0.12341725354588762, cov: 0.3043812061734312 }
var 14 with class: { cor: 0.30505546993959937, cov: 5.068416007551439 }
var 15 with class: { cor: 0.10394149892263876, cov: 0.22606579433657334 }
var 16 with class: { cor: 0.1022857469237104, cov: 25.891162989241394 }
var 17 with class: { cor: 0.057853582234509655, cov: 51.18752547560126 }
var 18 with class: { cor: 0.250037566644008, cov: 0.020614285270282647 }
var 19 with class: { cor: 0.2882745243177669, cov: 44.22791431783834 }
var 20 with class: { cor: 0.3104754379997538, cov: 68.94051184339085 }
var 21 with class: { cor: 0.2576824982043303, cov: 1513.1610567523849 }
```