Features

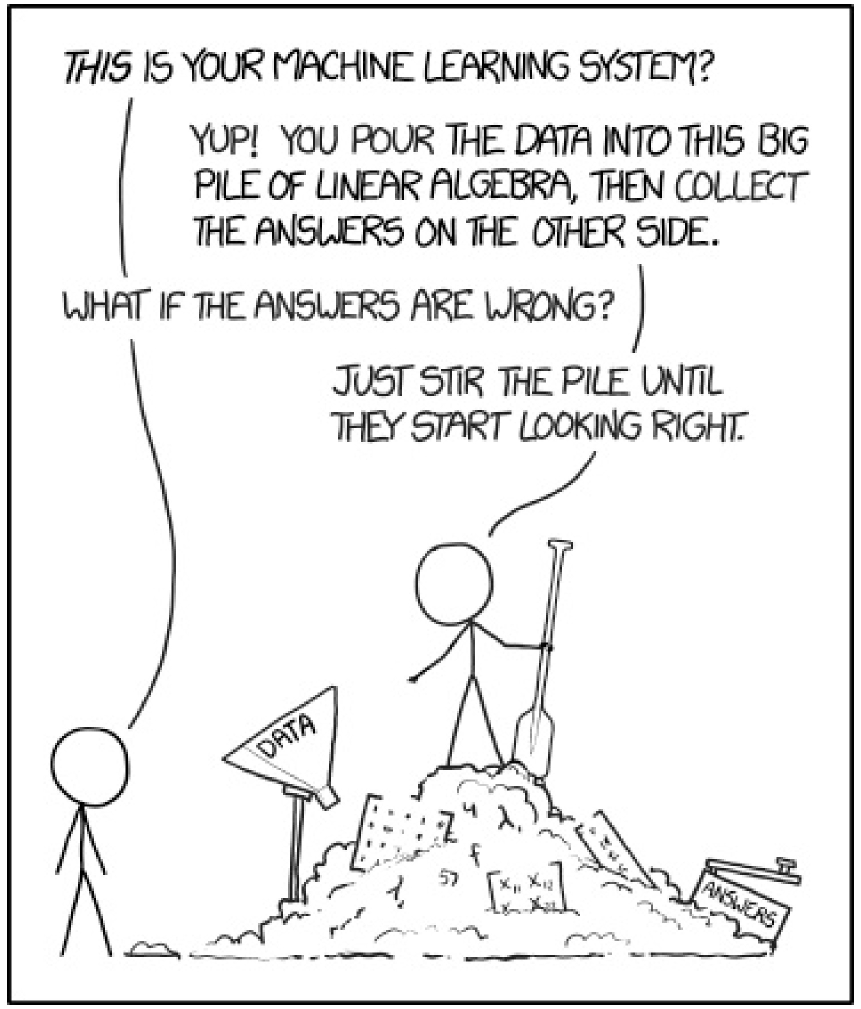
Applied Machine Learning with R

www.therbootcamp.com

@therbootcamp

January 2019

Feature issues



Too many features

- Curse of dimensionality
- Feature importance

Wrong features

- Feature scaling
- Feature correlation
- Feature quality

Create new features

- Feature engineering

Curse of dimensionality

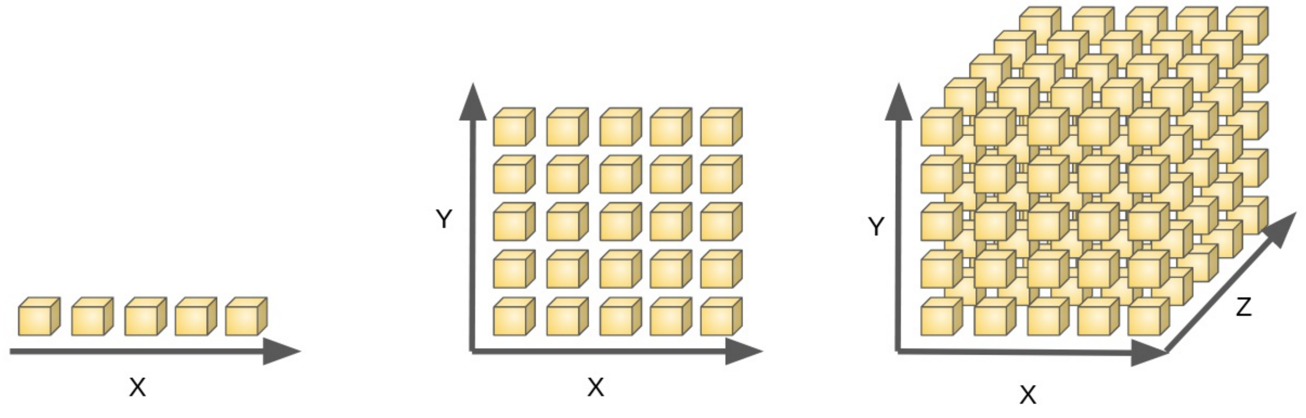
As the number of features grows…

Performance - the amount of data that needs to generalize accurately grows exponentially.

Efficiency - the amount of computations grows (how much depends on the model).

Redundancy - the amount of redundancy grows (how much depends on the model).

→ **Small set of good predictors**



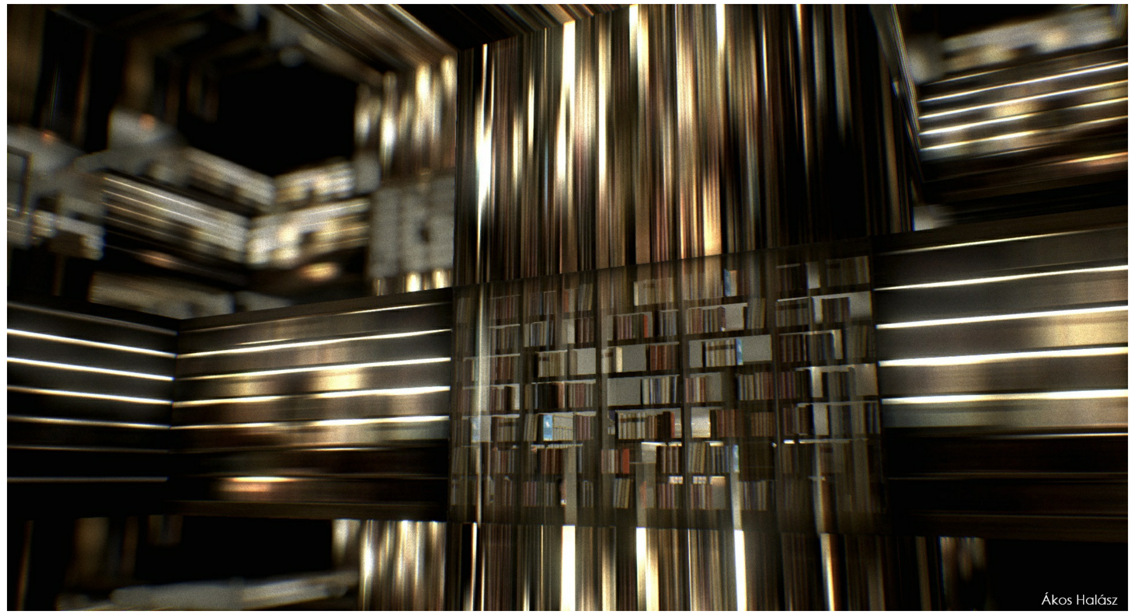
How to reduce dimensionality?

<u>3 ways</u>

Reduce variables **manually** based on statistical or intuitive considerations.

Reduce variables **automatically** using the right ML algorithms, e.g., `random forests` or `lasso regression`, or feature selection algorithms, e.g., `recursive feature selection`.

Compress variables using **dimensionality reduction algorithms**, such as `principal component analysis` (PCA).



Interstellar

Feature importance

Feature importance characterizes how much a feature contributes to the fitting/prediction performance.

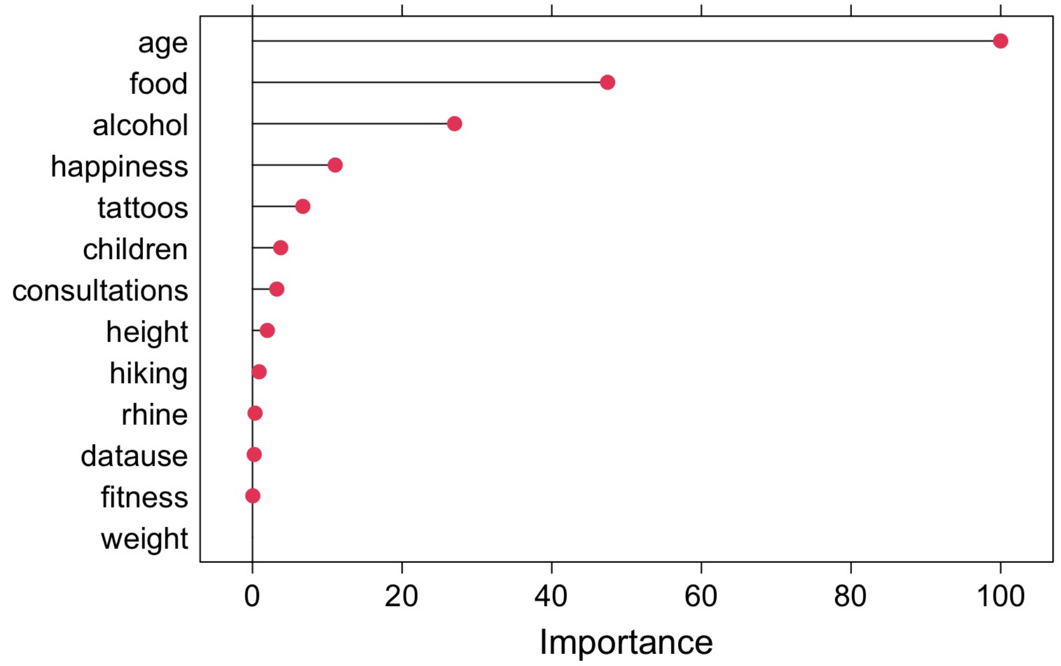
Typically **normalized** to `[0, 100]`.

There are many **model specific metrics**.

General strategies

- Single variable prediction (e.g., using LOESS, ROC)
- Accuracy loss from scrambling
- `random forests` importance
- etc.

```
# plot variable importance for lm(income ~ .)
plot(varImp(income_lm))
```



varImp()

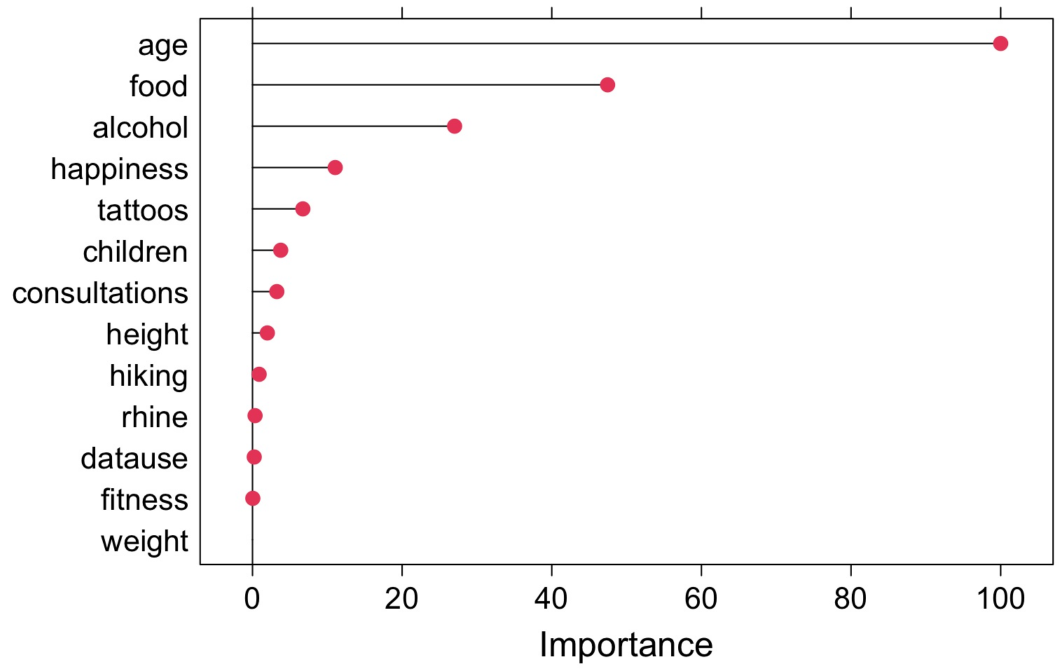
`varImp()` **automatically selects appropriate measure** of variable importance for a given algorithm.

```
varImp(income_lm)
```

```
lm variable importance

           Overall
age        100.000
food        47.714
alcohol     27.108
happiness   11.606
tattoos      7.243
children     4.060
height       1.748
datause      0.667
fitness      0.423
weight       0.000
```

```
# plot variable importance for lm(income ~ .)
plot(varImp(income_lm))
```



Recursive feature selection using `rfe()`

Algorithm(s) to **automatically select the best number of `n` predictors**, with `n` being selected from a set of candidate sets `N`, e.g., `N = [2,3,5,10]`, determined by the user.

Algorithm

1. **Resample** and split data
2. Identify **best `n` predictors** and their prediction performance
3. **Aggregate performance** and select best `n` and the accordingly best predictors

```
# Run feature elimination
rfe(x = ..., y = ...,
    sizes = c(3,4,5,10), # feature set sizes
    rfeControl = rfeControl(functions = lmFuncs))
```

```
Recursive feature selection

Outer resampling method: Bootstrapped (25 reps)

Resampling performance over subset size:

 Variables  RMSE Rsquared   MAE RMSESD RsquaredSD   MAESD Selected
         3 0.369    0.867 0.292 0.0120    0.01083 0.00990
         4 0.368    0.868 0.289 0.0117    0.01085 0.01006
         5 0.367    0.869 0.288 0.0106    0.00989 0.00921        *
        10 0.368    0.867 0.290 0.0116    0.01044 0.00967
        14 0.369    0.867 0.290 0.0116    0.01029 0.00987

The top 5 variables (out of 5):
   age, food, alcohol, happiness, tattoos
```

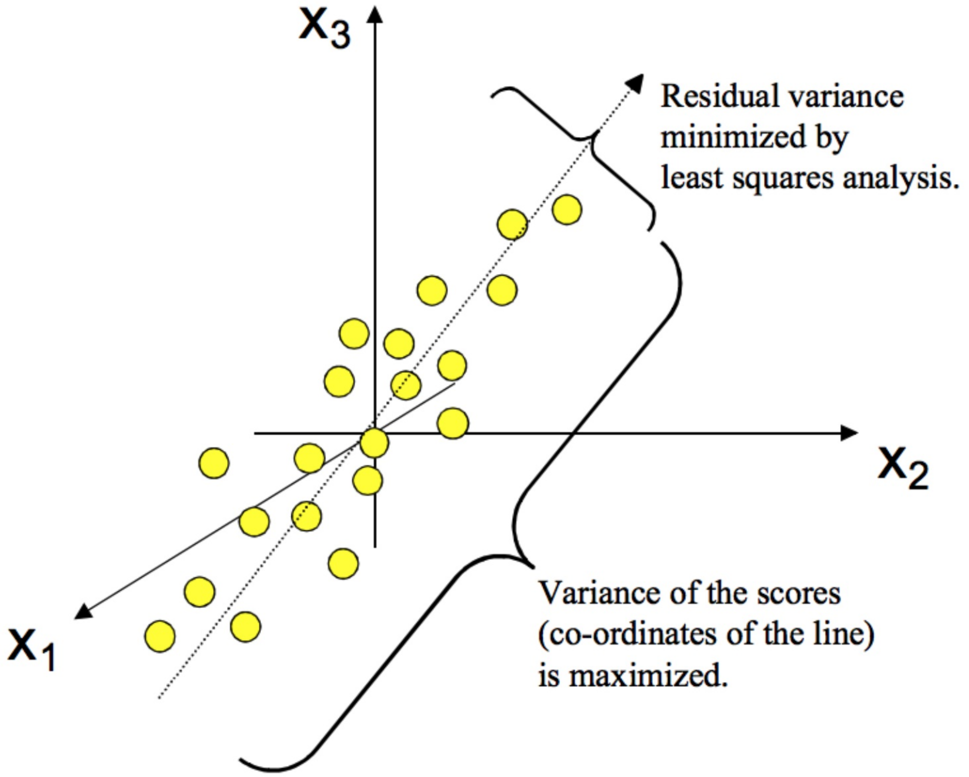
Dimensionality reduction using PCA

The go-to algorithm for dimensionality is **principal component analysis** (PCA).

PCA is an **unsupervised**, **regression-based** algorithm that re-represents the data in a **new feature space**.

The new features aka **principal components are greedy** in that they attempt to explain as much variance as they can leaving as little as possible to other components.

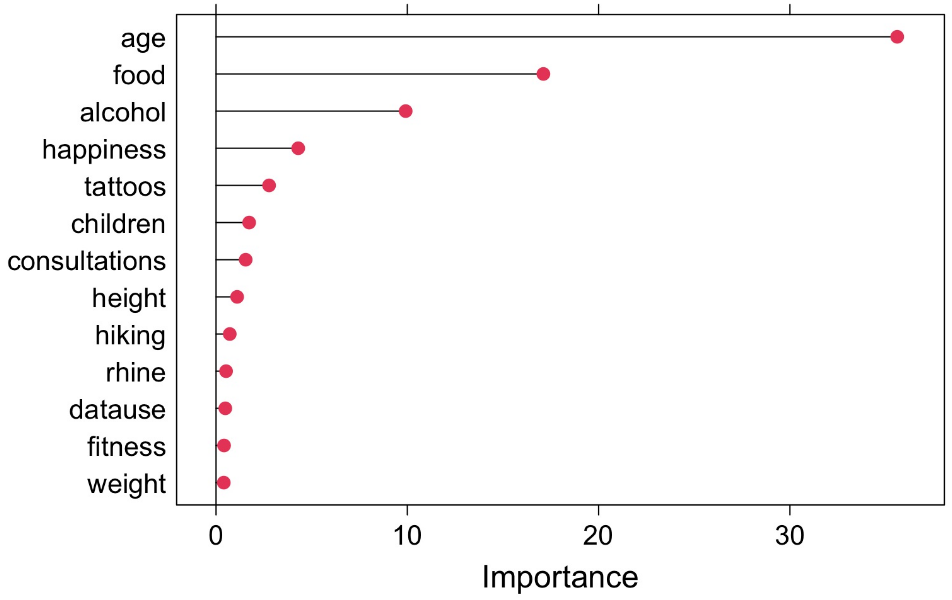
Skimming the best components off the top results in a small number of features that **preserve the original features as well as possible**.



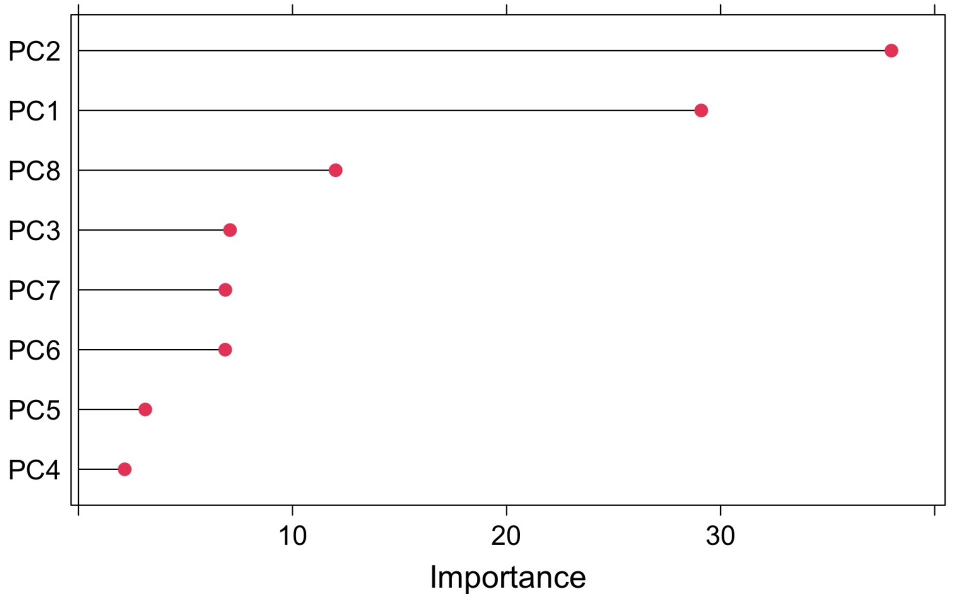
Using PCA

```
# train model WITHOUT PCA preprocessing
model = train(income ~ ., method = 'lm',
              data = bas_train)

plot(varImp(model))
```




```
# train model WITH PCA preprocessing
model = train(income ~ ., method = 'lm',
                 data = bas_train,
                 preProc = c('pca'))
plot(varImp(model))
```



Other, easy feature problems

Multi-collinearity

Multi-collinearity, **high feature correlations**, mean that there is redundancy in the data, which can lead to **less stable fits**, **uninterpretable variable importances**, and **worse predictions**.

```
# identify redundant variables
findCorrelation(cor(baselers))
```

```
[1] 5
```

```
# remove from data
remove <- findCorrelation(cor(baselers))
baselers <- baselers %>%
  select(-remove)
```

Unequal & low variance

Unequal variance **breaks regularization** (L1, L2) and renders estimates difficult to interpret.

```
# standardize and center variables
train(..., preProc("center", "scale"))
```

Low variance variables add parameters, but **can hardly contribute to prediction** and are, thus, also redundant.

```
# identify low variance variables
nearZeroVar(baselers)
```

```
integer(0)
```

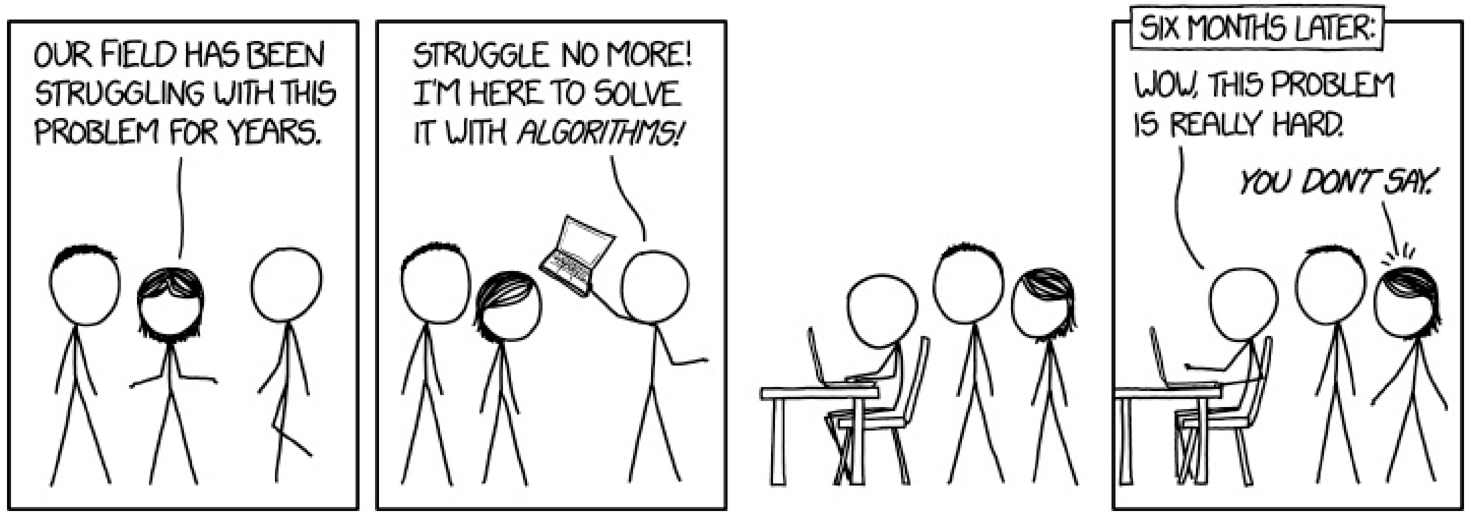
Difficult feature problems

1 **Trivial features**

Successful prediction not necessarily implies that a meaningful pattern has been detected.

2 **Missing features**

Some problems are hard, requiring the engineering of new features.

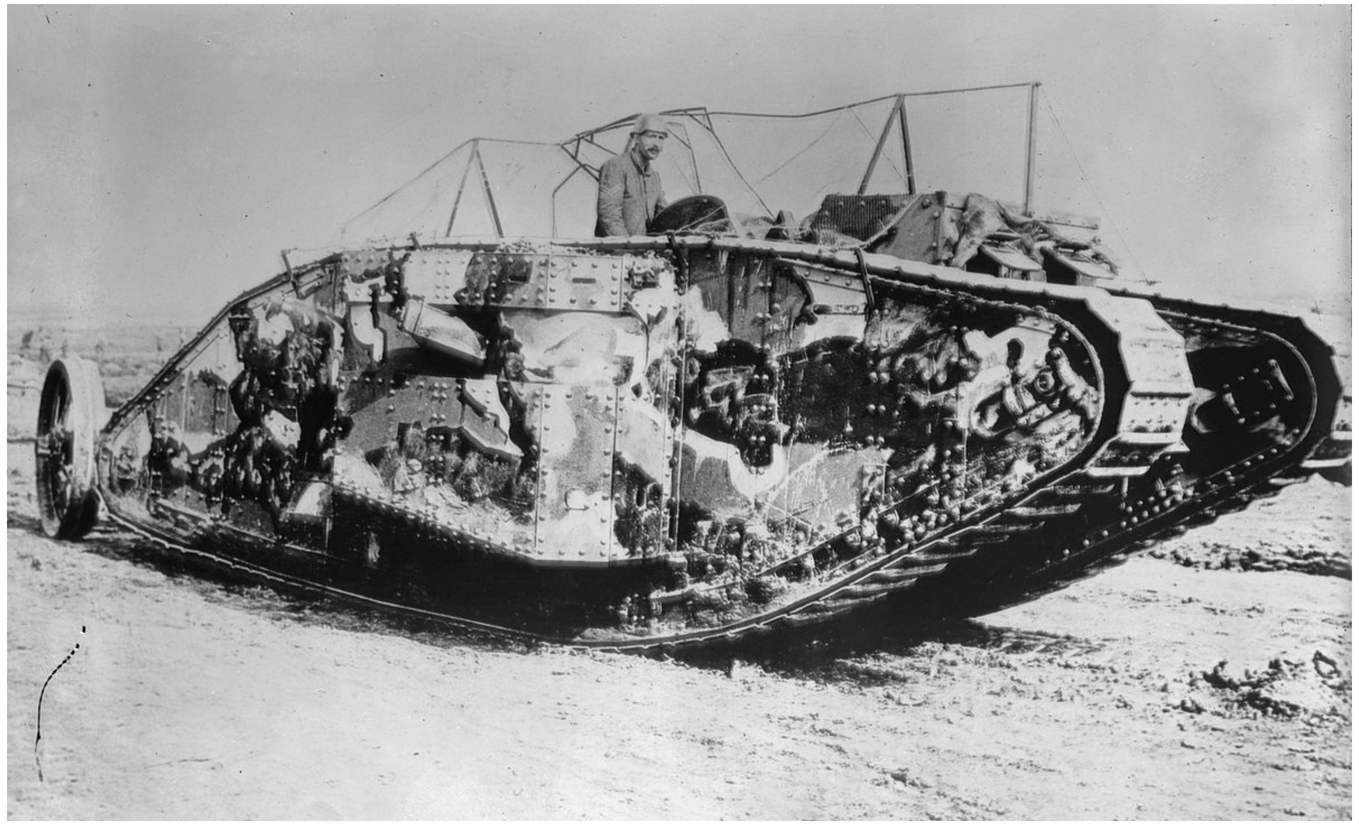


Trivial features

An urban myth?!

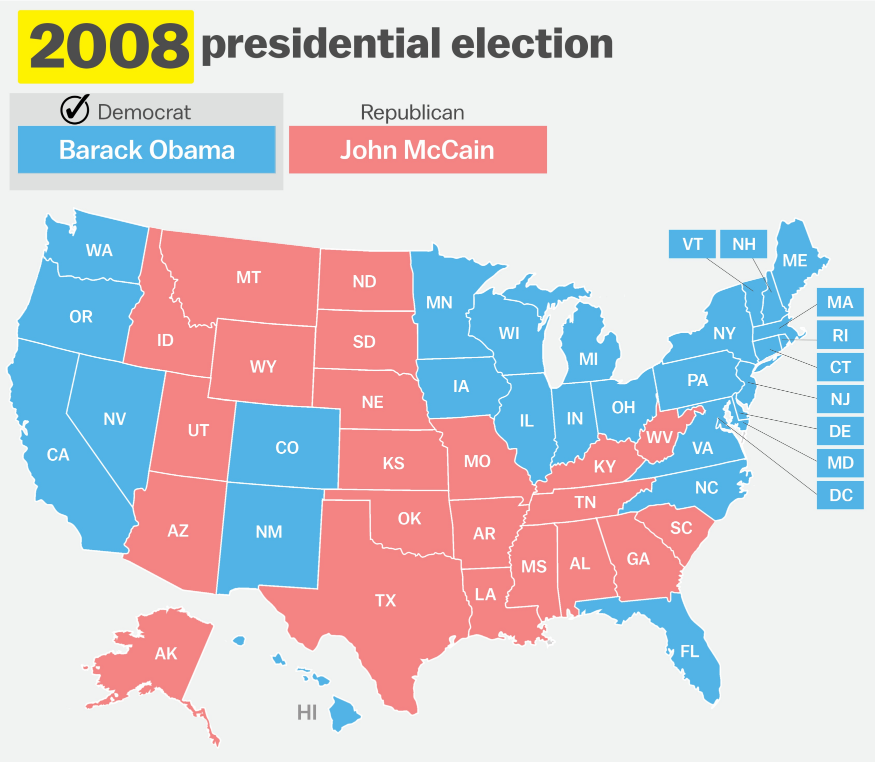
"The Army trained a program to differentiate American tanks from Russian tanks with 100% accuracy. Only later did analysts realize that the American tanks had been photographed on a sunny day and the Russian tanks had been photographed on a cloudy day. The computer had learned to detect brightness."

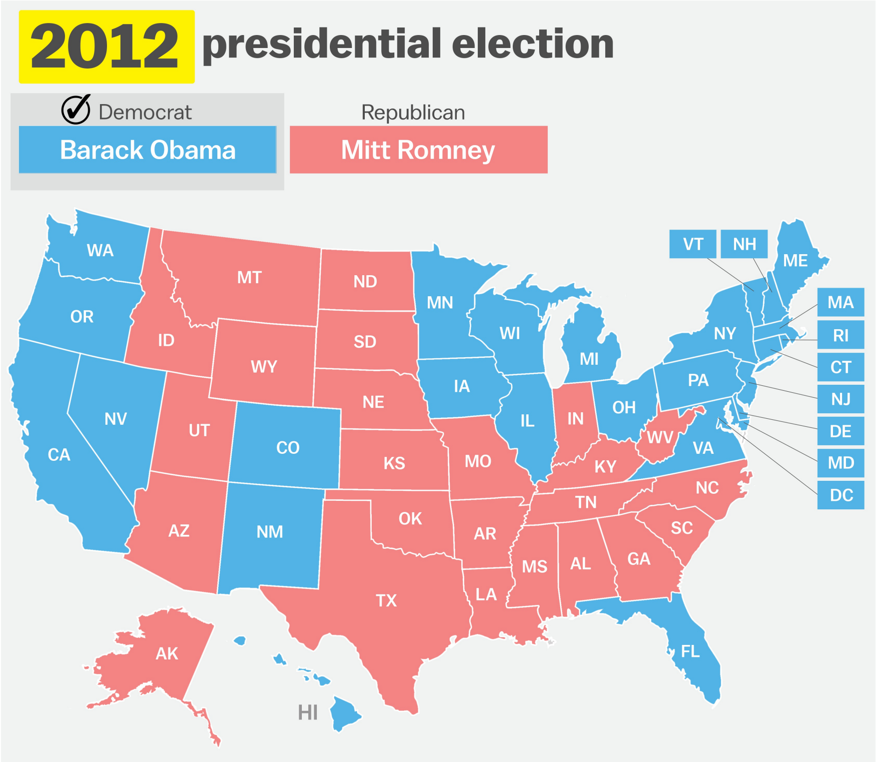
New York Times [Full text]



Trivial features

In 2012, Nate Silver was praised to have correctly predicted the outcomes of the presidential election in 50 states after having correctly predicted 49 states in 2009. **But how much of a challenge was that?**





(Always!) missing features

Feature Engineering

Pedro Domingos

Xavier Conort

Andrew Ng

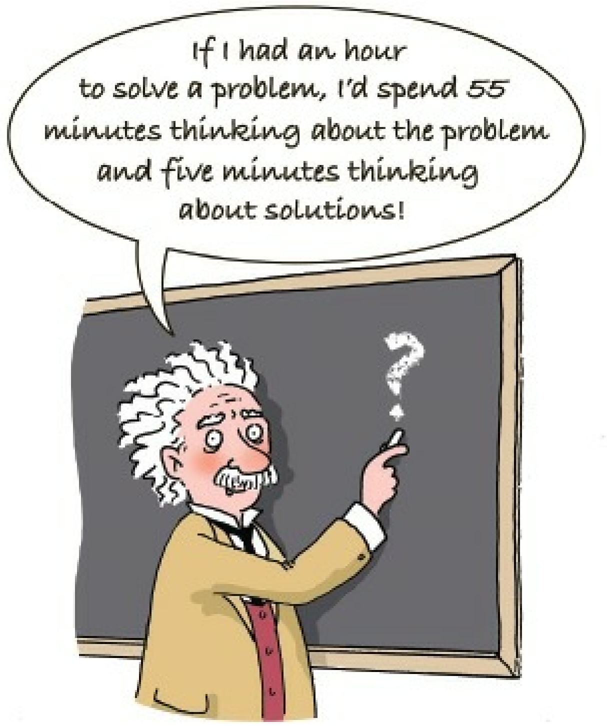
Feature engineering

Jason Brownlee

duw

Feature engineering involves

- **Transformations**
- **Interactions**
- **New features**



Practical