

Building meaningful machine learning models for disease prediction

Dr Shirin Glander

Dep. of Genetic Epidemiology
Institute of Human Genetics
University of Münster

shirin.glander@wwu.de

<https://shiring.github.io>
<https://github.com/ShirinG>

Friday, 31st March 2017

Dr Shirin Glander

Dep. of Genetic Epidemiology
Institute of Human Genetics
University of Münstershirin.glander@uni-muenster.de<https://shirlog.github.io>
<https://github.com/ShirinG>Friday, 31st March 2017

- Welcome everybody!
- Thank you very much for the invitation and the opportunity to present this talk
- I will go over my work on using machine learning models in disease prediction
- I want to specifically give a hands-on demonstration of how you can build meaningful models yourselves using R
- I will demonstrate how to evaluate model performance and
- how to optimize models to address different disease-related questions

About me

since 2015 Bioinformatics Postdoc
Next Generation Sequencing
autoinflammatory diseases &
innate immunity

2011 - 2015 PhD in Biology
Is the immune system of plants required to adapt to
flowering time change?

2005 - 2011 BSc and MSc of Science in Biology
evolutionary genetics,
immune memory in *Drosophila*



About me

About me



since 2015 Bioinformatics Postdoc
Next Generation Sequencing
autoinflammatory diseases &
innate immunity

2011 - 2015 PhD in Biology
Is the immune system of plants required to adapt to
flowering time change?

2005 - 2011 BSc and MSc of Science in Biology
evolutionary genetics,
immune memory in *Drosophila*

- Before I start, I want to quickly introduce myself:
- I am a bioinformatics postdoc
- working with next generation sequencing data,
- like RNA-seq for transcriptomics,
- whole genome sequencing for variant analysis,
- ATAC- or Chip-seq for chromatin and epigenetic information.
- My own research focuses on questions relating to autoinflammatory diseases
- and innate immune mechanisms

- I earned my PhD in biology from the University of Münster in 2015
- working with RNA-seq data to determine how plant defense has co-evolved with and potentially shaped different life-history strategies

- Before that, during my BSc and MSc I worked on questions about evolutionary genetics and immune memory in *Drosophila*

Table of contents

Building meaningful machine learning models for disease prediction

- 1 What makes a model meaningful?
- 2 Machine Learning (ML) in disease modeling
- 3 A quick recap of ML basics
- 4 How to build ML models in R
- 5 Evaluating model performance

Table of contents

Table of contents

Building meaningful machine learning models for disease prediction

- 1 What makes a model meaningful?
- 2 Machine Learning (ML) in disease modeling
- 3 A quick recap of ML basics
- 4 How to build ML models in R
- 5 Evaluating model performance

- Now, let's start with the topic about which you're here:
- Machine learning is a powerful approach for developing sophisticated, automatic, and objective models for the analysis of complex biomedical data
- Machine learning promises to help physicians make near-perfect diagnoses, ...
- ... choose the best medications for their patients, ...
- ... predict readmissions, ...
- ... identify patients at high-risk for poor outcomes, ...
- ... and in general improve patients' health while minimizing costs.
- I titled my talk: "Building meaningful machine learning models for disease prediction"
- with the emphasis on **meaningful**
- So, first, I want to introduce to you what it is I mean exactly with **meaningful models**
- Then, I will give you a few examples of how machine learning is currently being used in disease modeling and clinical data science
- Before I delve into the nitty-gritty of modeling, I will quickly recap the most important concepts of ML
- And finally, I will show how to build ML models in R
- and how to evaluate the performance of such models

What makes a model meaningful?

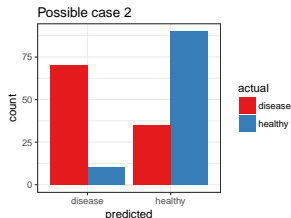
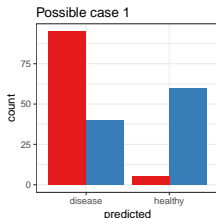
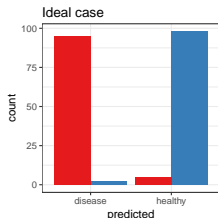
What makes a model meaningful?

- creating ML models is relatively easy
- creating **good or meaningful** models is hard

Meaningful models

- are generalizable
- answer the question(s) posed...
- ... with sufficient accuracy to be trustworthy

Accuracy depends on the problem!



What makes a model meaningful?

What makes a model meaningful?

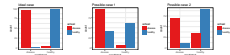
What makes a model meaningful?

- creating ML models is relatively easy
- creating *good or meaningful* models is hard

Meaningful models

- are generalizable
- answer the question(s) posed...
- ... with sufficient accuracy to be trustworthy

Accuracy depends on the problem!



- With a little bit of practice, anybody can build machine learning models
 - what is hard though, is to build 'good' or 'meaningful' models
 - I want to begin with an introduction to the main question of my talk:
 - what makes a model *good* or *meaningful*?
- it needs to be generalizable, i.e. it needs to perform well on unseen data
 - It answers a **specific** question or addresses a specific problem, e.g. does a mammogram image show a healthy breast or is there a tumor? Or does an ECG show a normal heart rhythm or arrhythmia?
 - And it produces a correct outcome (e.g. a diagnosis) often enough that we trust it!
- If we build models, we therefore need to evaluate its accuracy
 - before we can decide whether it is trustworthy enough to implement in real-life,
 - like in a hospital where it could e.g. assist doctors in making decisions on treatment
 - But what exactly is *high accuracy* can not be defined with a one-size-fits-all approach:
 - it depends on the problem we want to model.
 - a model needs to perform well in the real world, not in the lab (i.e. on training data)
 - just because a model performs well in the lab, doesn't mean it will also perform well in deployment
 - the context might change: distribution, missing data, noise, class size, etc.

What makes a model meaningful?

What makes a model meaningful?

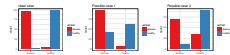
What makes a model meaningful?

- creating ML models is relatively easy
- creating **good or meaningful** models is hard

Meaningful models

- are generalizable
- answer the question(s) posed...
- ... with sufficient accuracy to be trustworthy

Accuracy depends on the problem!



Let me illustrate what I mean with the following examples:

- Ideal case:** Of course, we all want to achieve ideal modeling results where overall prediction accuracy is very high. With a model like that, we can be very confident that a healthy person is indeed healthy and a sick person is not.
- But in reality, we often achieve prediction accuracies that are much less nice.
- Scenarios 1 and 2:** Let's consider two possible scenarios:
 - in scenario 1, we can be very confident that a person who got classified as "healthy" is indeed healthy, while a person who has been diagnosed as diseased might as well be healthy based on these prediction accuracies
 - in case 2, it is the other way around.
- We now need to make a decision which scenario is better and in which direction we want to optimize our model: do we rather want to refer a few healthy people for further checking because the model predicted them as diseased? Or do we rather want to be as certain as possible that a predicted disease is actually true and accept that we might miss a few disease cases?

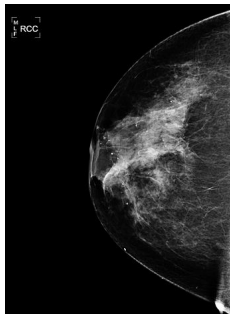
Machine Learning (ML) in disease modeling

ML in disease modeling

- tools that can interpret "big medical data"
- and provide fast, accurate and actionable information
- for precision or personalized medicine

Examples:

- computer-aided diagnosis of breast cancer from mammograms¹
- identifying gene defects with facial recognition software²
- identifying signatures of Brain Cancer from MRSI³
- ... and many more ...



¹Doi 2007.

²Levenson 2014.

³Sadja 2006.

Image source: Wikimedia Commons

Machine Learning (ML) in disease modeling

ML in disease modeling

- tools that can interpret "big medical data"
- and provide fast, accurate and actionable information
- for precision or personalized medicine

Examples:

- computer-aided diagnosis of breast cancer from mammograms¹
- identifying gene defects with facial recognition software²
- identifying signatures of Brain Cancer from MRSI³
- ... and many more ...



¹Dai 2007,
²Lawrence 2014,
³Siege 2006.

- there is not a place in medicine where AI doesn't have potential applications:
- more data is being collected that needs to be interpreted
- increasingly data driven diagnosis, e.g. in radiology
- image recognition of e.g. tumors or pneumonia in medical images
- similar to training ML models to recognize images of cats (classification in images)
- ML allows incorporation of data from heterogenous inputs:
- clinical, genomics, drugs, electronic health records, etc.
- = personalised medicine
- A key aspect of precision medicine is the development of informatics tools
- that can analyze and interpret 'big data' sets
- in an automated and adaptive fashion
- while providing accurate and actionable clinical information
- ML based models can improve detection, diagnosis, and therapeutic monitoring of disease
- you can find things with ML that you wouldn't be able to find otherwise
- many models today perform better than humans!
- a patient with a rare difficult condition can be matched to similar cases from the past
- faster diagnosis, better treatment

Machine Learning (ML) in disease modeling

ML in disease modeling

- tools that can interpret "big medical data"
- and provide fast, accurate and actionable information
- for precision or personalized medicine

Examples:

- computer-aided diagnosis of breast cancer from mammograms¹
- identifying gene defects with facial recognition software²
- identifying signatures of Brain Cancer from MRSI³
- ... and many more ...

¹Dai 2007.
²Lewinstein 2014.
³Siege 2005.



Image source: Wikimedia Commons

- identifying breast cancer, lung cancer, osteoporosis, brain tumors, etc. from medical images
- predicting response of different cancer types to different treatments
- Computer-Aided Diagnosis (CAD) has become one of the major research subjects in medical imaging and diagnostic radiology
- With CAD, radiologists use the computer output as a 'second opinion' and make the final decisions
- In vivo magnetic resonance spectroscopy imaging (MRSI) allows noninvasive characterization
- and quantification of molecular markers of potentially high clinical utility
- for improving detection, identification, and treatment for a variety of diseases, most notably brain cancers
- facial analysis technology aids diagnoses of genetic disorders
- researchers at the University of Oxford in the United Kingdom
- Face2Gene uses machine learning
- helps geneticists narrow down possible disorders that often involve dysmorphic facial features
- doctors are still important!
- computer does the tasks that we humans are not good at, like interpreting complex images
- we have more data than humans can manage to make sense of (e.g. genomics data)
- the clinicians will be freed up to think about best treatment options and talk more with the patients
- good models are built on strong knowledge of the question and the biology behind it
- features need to be evaluated in context

Can we trust a model?



- most ML algorithms model high-degree interactions between variables
- we often don't know **WHY** ML models make decisions
- inherent problem with ML models: they are hard (or impossible) to interpret
- therefore, it is crucial that our models are **meaningful**

Image source: Pixabay



- most ML algorithms model high-degree interactions between variables
- we often don't know **WHY** ML models make decisions
- inherent problem with ML models: they are hard (or impossible) to interpret
- therefore, it is crucial that our models are **meaningful**

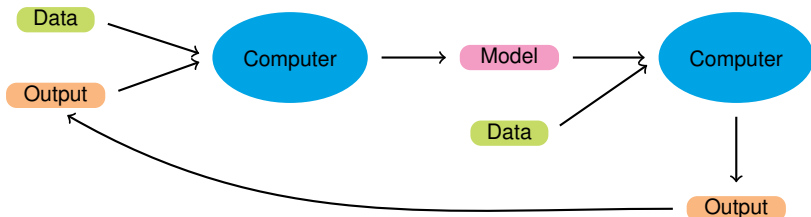
Image source: Pixabay

- the elephant in the room
- machine learning algorithms tend to create nonlinear,
- non-monotonic,
- non-polynomial,
- and even non-continuous functions
- that approximate the relationship between independent and dependent variables in a data set
- most ML algorithms model high-degree interactions between variables (meaning the effect of combining many?i.e., more than two or three?variables together)
- When working with data you should be asking yourselves some very hard questions:
 - do I understand my data?
 - Do I understand the model and answers my machine learning algorithm is giving me?
 - And do I trust these answers?
- Unfortunately, the complexity that bestows the extraordinary predictive abilities on machine learning algorithms also makes the answers the algorithms produce hard to understand,
- and maybe even hard to trust.
- almost by definition one cannot really understand a good model
- difficulties in interpretation still present a barrier for the widespread, practical use of ML
- trusting models and results is a general requirement for good (data) science
- we want to understand them as well as possible

A quick recap of ML basics

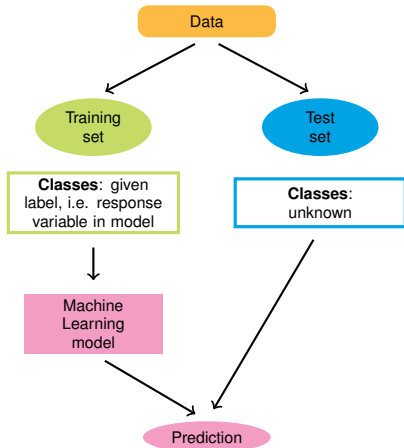
Machine learning

- artificial intelligence (AI)
- data-driven
- algorithms **learn** by being trained on observed data...
- ... and **predict unknown data**
- ML concepts are not new, but the increase in computational capacity has made them more accessible

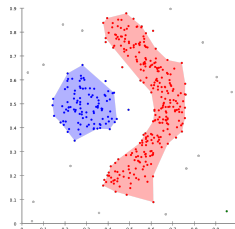
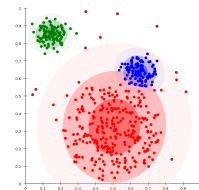


Supervised vs Unsupervised algorithms

Supervised



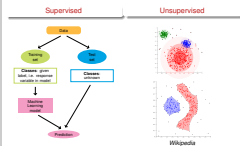
Unsupervised



Wikipedia

A quick recap of ML basics

Supervised vs Unsupervised algorithms

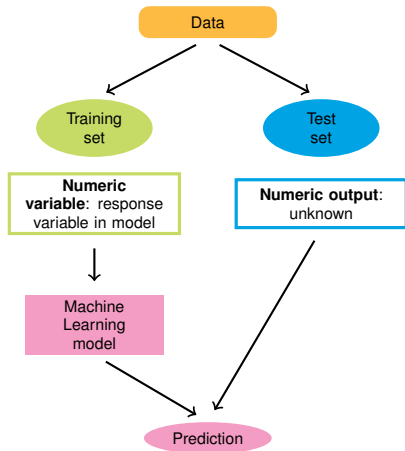


- Unsupervised:
 - no input!
 - matrix decomposition methods, e.g. nonnegative matrix factorization
 - pattern recognition
 - clusters data into different groups
 - able to recover biomarkers of disease and toxicity
- Supervised
 - classification labels are given
 - e.g. support vector machine
- Semi-supervised:
 - a little bit of input, like how many classes to find

Here, I will focus on supervised learning!

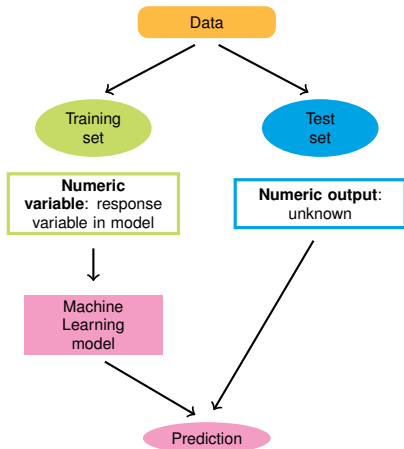
Classification vs Regression

Regression
e.g. weight loss

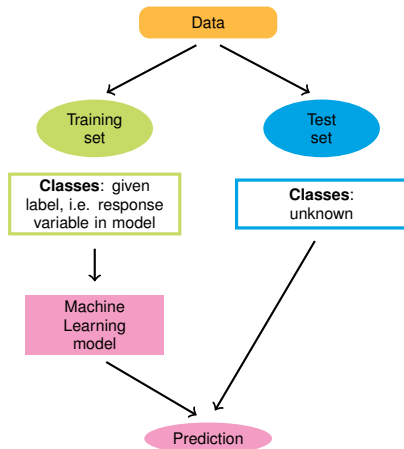


Classification vs Regression

Regression
e.g. weight loss

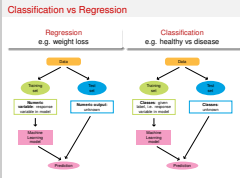


Classification
e.g. healthy vs disease



A quick recap of ML basics

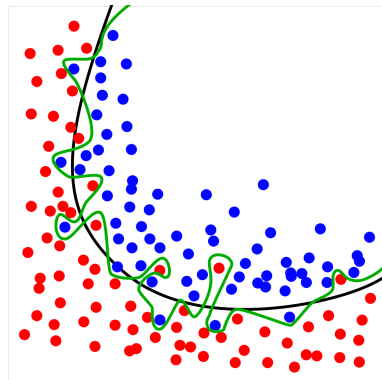
Classification vs Regression



- Supervised learning!
 - Your learning algorithm seeks a function from inputs to the respective targets.
 - If the targets are expressed in some classes, it is called classification problem.
 - Alternatively, if the target space is continuous, it is called regression problem.
-
- Functions created by linear regression algorithms are probably the most interpretable class of models.
 - For this reason, linear models were the go-to applied predictive modeling tool for decades,
 - even though it usually meant giving up a couple points on the accuracy scale.
 - These models will be referred to here as 'linear and monotonic',
 - meaning that for a change in any given independent variable, the response function changes at a defined rate,
 - in only one direction, and at a magnitude represented by a readily available coefficient
-
- Most machine learning algorithms create nonlinear, non-monotonic response functions.
 - This class of functions is the most difficult to interpret, as they can change in a positive and negative direction and at a varying rate for any change in an independent variable.
 - Typically, the only standard interpretability measure these functions provide are relative variable importance measures

Features

- Features are the variables used for model training.
- Using the right features is crucial.
- More is not necessarily better (overfitting)!
- feature selection
- feature extraction/ engineering



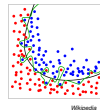
Wikipedia

A quick recap of ML basics

Features

Features

- ◆ Features are the variables used for model training.
- ◆ Using the right features is crucial.
- ◆ More is not necessarily better (overfitting)!
- ◆ feature selection
- ◆ feature extraction/ engineering



- Machine learning uses so called features (i.e. variables or attributes) to generate predictive models
- Using a suitable combination of features is essential for obtaining high precision and accuracy.
- Because too many (unspecific) features pose the problem of overfitting the model, we generally want to restrict the features in our models to those, that are most relevant for the response variable we want to predict.
- Using as few features as possible will also reduce the complexity of our models, which means it needs less time and computer power to run and is easier to understand.
- Sometimes extraction of salient structure in the data that is more informative than the raw data itself (the feature extraction problem)

Training, (cross-) validation and test data

└ A quick recap of ML basics

└ Training, (cross-) validation and test data

Decide cross validation strategy - To avoid overfitting, make sure you've set up a cross validation strategy in early stages. A nice CV strategy will help you get reliable score on leaderboard.

preventing overfitting, we want our models to be generalizable

Cross validation means that from my main set, I create RANDOMLY 2 sets. I built (train) my algorithm with the first one (let's call it training set) and score the other (let's call it validation set). I repeat this process multiple times and always check how my model performs on the test set in respect to the metric I want to optimize.

The process may look like:

For 10 (you choose how many X) times Split the set in training (50% And validation (50% Then fit the algorithm on the training set Score the validation set. Save the result of that scoring in respect to the chosen metric. Calculate the average of these 10 (X) times. That how much you expect this score in real life and is generally a good estimate.

Remember to use a SEED to be able to replicate these X splits Other things to consider is Kfold and stratified KFold .

Read here. For time sensitive data, make certain you always the rule of having past predicting future when testings.

Hyper Parameter Tuning

Grid Search

Start hyper parameter tuning - Once CV is at place, try improving model's accuracy using hyper parameter tuning. It further includes the following steps: Data transformations: It involve steps like scaling, removing outliers, treating null values, transform categorical variables, do feature selections, create interactions etc. Choosing algorithms and tuning their hyper parameters: Try multiple algorithms to understand how model performance changes. Saving results: From all the models trained above, make sure you save their predictions. They will be useful for ensembling. Combining models: At last, ensemble the models, possibly on multiple levels. Make sure the models are correlated for best results.

Take home messages:

- ...

How to build ML models in R

Session setup

- Breast Cancer Wisconsin Dataset⁴



- caret⁵
- h2o⁶

Code will be available on [my website](#) and on [Github](#)

⁴W. H. Wolberg and O. L. Mangasarian (1990). “Multisurface method of pattern separation for medical diagnosis applied to breast cytology.” In: *Proceedings of the National Academy of Sciences* 87.23, pp. 9193–9196.

⁵M. Kuhn et al. (2016). *caret: Classification and Regression Training*. R package version 6.0-71.

⁶H2O.ai (2017). *h2o: R Interface for H2O*. R package version 3.10.3.6.

How to build ML models in R

Session setup

Breast Cancer Wisconsin Dataset⁴

■ caret⁵
 ■ h2o⁶

Code will be available on [my website](#) and on [Github](#)

⁴W. H. Wolberg and O. L. Mangiswaran (1990). "Multisurface method of pattern separation for medical diagnosis applied to breast cytology." In: *Proceedings of the National Academy of Sciences* 87:23, pp. 9193-9196.

⁵M. Kuhn et al. (2016). *caret: Classification and Regression Training*. R package version 6.0-71.

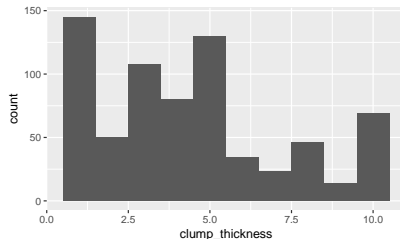
⁶H2O.ai (2017). *H2O: R Interface for H2O*. R package version 3.10.3.6.

- This breast cancer databases was obtained from the University of Wisconsin Hospitals, Madison
- from Dr. William H. Wolberg
- It looks at the predictor classes:
 - malignant or
 - benign breast mass
- The features characterise cell nucleus properties
 - and were generated from image analysis of fine needle aspirates (FNA) of breast masses:
 - Sample ID (code number)
 - Clump thickness
 - Uniformity of cell size
 - Uniformity of cell shape
 - Marginal adhesion
 - Single epithelial cell size
 - Number of bare nuclei
 - Bland chromatin
 - Number of normal nuclei
 - Mitosis
 - Classes, i.e. diagnosis

Get to know your data

Response variable

- Is it balanced?



Missing data

- Is there missing data?
- Can we afford to loose data points with missing values?
- Or do we use imputation (and introduce additional uncertainty)?

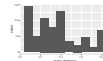
How to build ML models in R

Get to know your data

Get to know your data

Response variable

Is it balanced?



Missing data

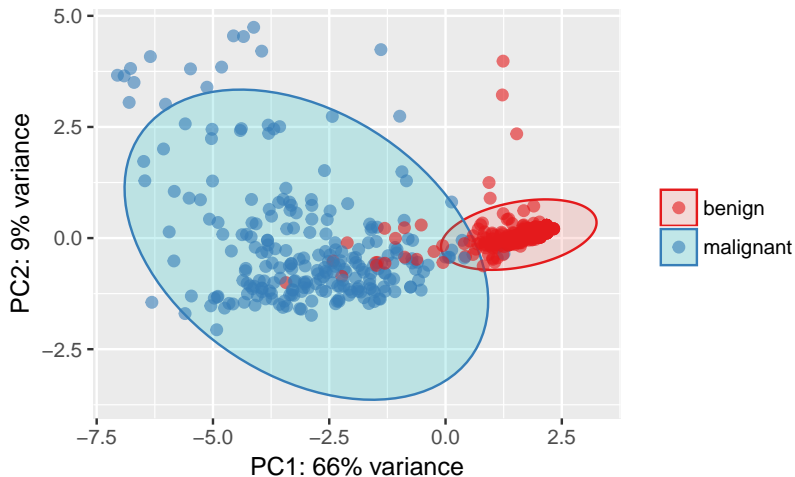
Is there missing data?

- Can we afford to loose data points with missing values?
- Or do we use imputation (and introduce additional uncertainty)?

- Understand the data
- Look at data types.
- Check variable classes.
- Create some univariate - bivariate plots to understand the nature of variables.
- Most important to know is the distribution: are the classes balanced?
- If we have unbalanced data, this will effect our model later on
We would have to consider over- or undersampling.
- Missing data
 - If we have lots of data and few missing values, we can afford to loose these data points.
 - If we don't have that much data though, our model will probably loose significant power if we remove the samples.
 - In that case, we would rather introduce a certain uncertainty by imputing missing values.

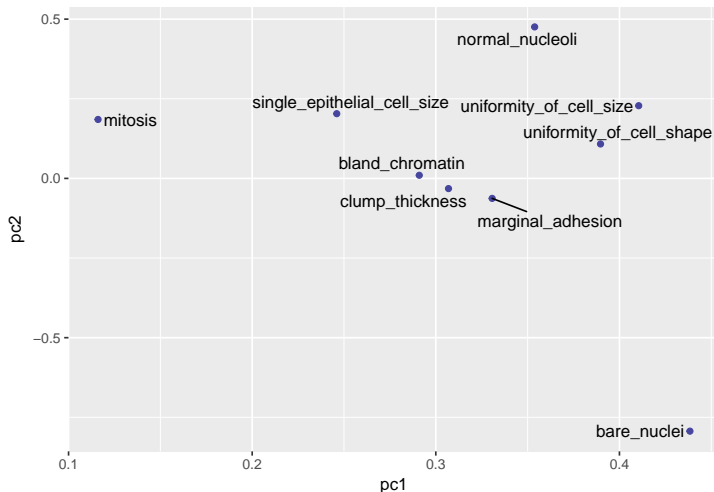
Get to know your data

Principal Component Analysis (PCA)



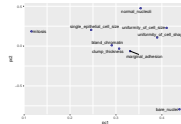
Get to know your data

Principal Component Analysis (PCA)



How to build ML models in R

Get to know your data

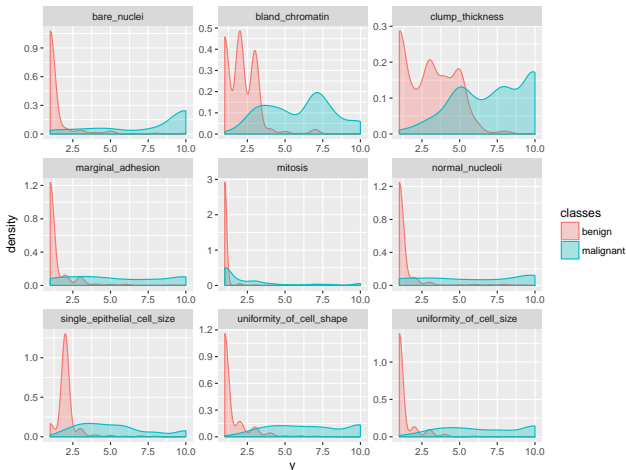


- Most real data sets are hard to see because they have many variables and many rows
 - visualization can help when we reduce the complexity
 - There are many techniques for projecting the rows of a data set from a usually high-dimensional original space into a more visually understandable lower-dimensional space:
 - Principal Component Analysis (PCA)
 - Multidimensional Scaling (MDS)
 - t-distributed Stochastic Neighbor Embedding (t-SNE)
 - Each of these techniques has strengths and weaknesses, but the key idea they all share is to represent the rows of a data set in a meaningful low-dimensional space.
-
- To get an idea about the dimensionality and variance of the datasets, I am first looking at PCA plots
 - for samples and features
 - Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation ...
 - ... to convert a set of observations into a set of values of linearly uncorrelated variables called principal components
 - The first two principal components (PCs) show the two components that explain the majority of variation in the data

Get to know your data

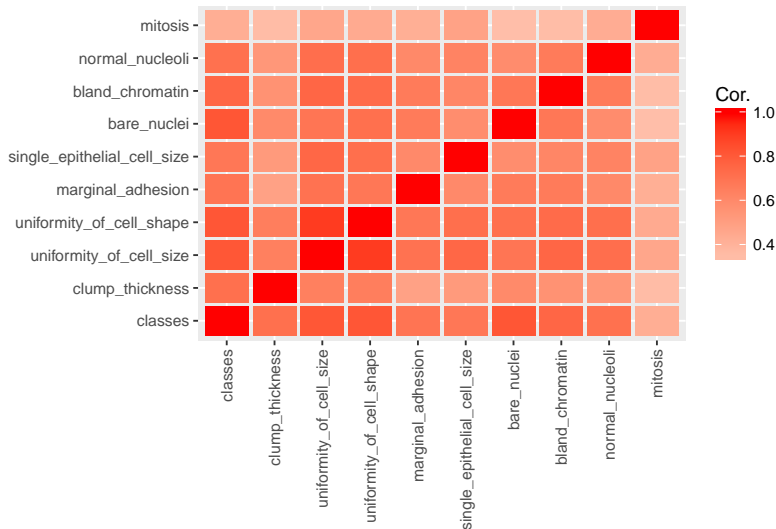
Features

- factors or numeric
- pre-processing



Get to know your data

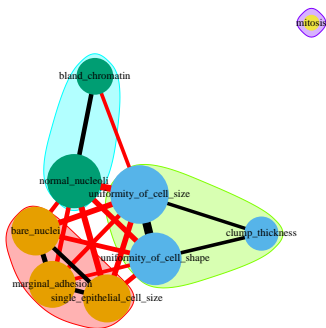
Correlation



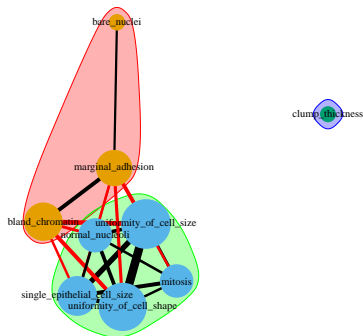
Get to know your data

Correlation graphs

Benign tumors



Malignant tumors



2017-03-27

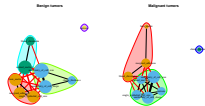
Webinar for ISDS R Group

How to build ML models in R

Get to know your data

Get to know your data

Correlation graphs

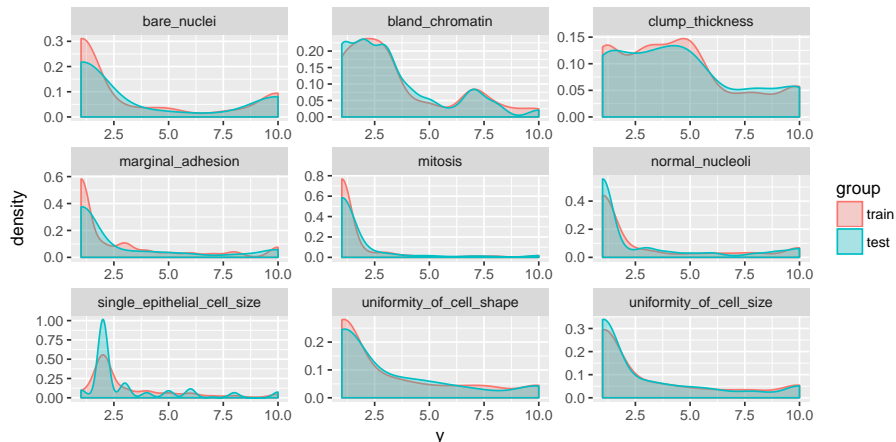


- A correlation graph is a two-dimensional representation of the relationships (correlation) in a data set
- Pearson correlation coefficient between features
- node size: sum of correlation coefficients
- edge width: correlation coefficient

Training, validation and test data

We need to split the data into training and test sets - ideally **stratified** by response class.

Density distribution



2017-03-27

Webinar for ISDS R Group

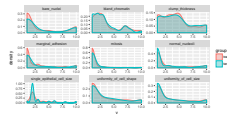
How to build ML models in R

Training, validation and test data

Training, validation and test data

We need to split the data into training and test sets - ideally *stratified* by response class.

Density distribution

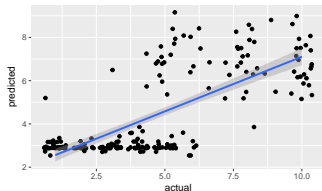


For accurate predictions, density distribution should be similar in training and test data!

Model examples

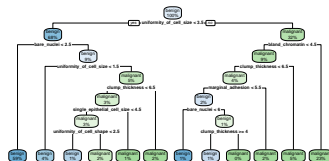
Regression with Linear Models

- e.g. Generalized Linear Models
- with *caret*



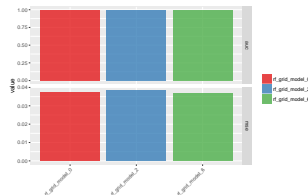
Tree-based classification

- Random Forest or Gradient boosting trees
- with *caret*



Hyper-parameter tuning

- Grid Search
- with *h2o*



How to build ML models in R

Model examples

Model examples

Regression with Linear Models

- e.g. Generalized Linear Models
- with caret



Tree-based classification

- Random Forest or Gradient boosting trees
- with caret



Hyper-parameter tuning

- Grid Search
- with h2o



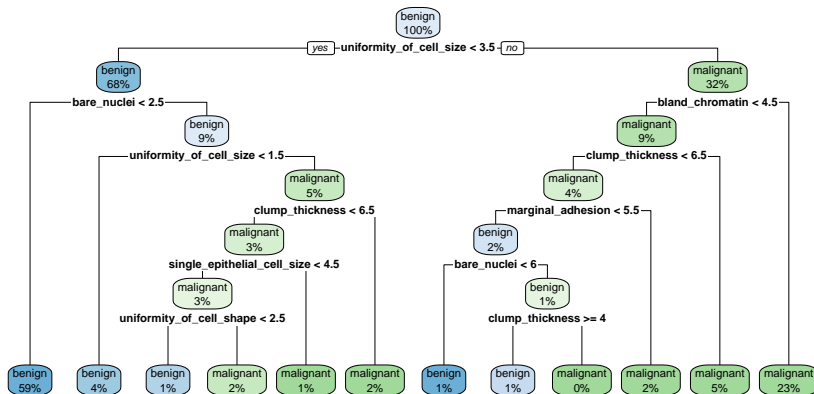
There are over 250 ML algorithms implemented with caret alone. I am going to focus on two of the most widely used.

- traditional linear models tend to create linear, monotonic, and continuous functions
- Even though they're not always the most accurate predictors, the elegant simplicity of linear models makes the results they generate easy to interpret.
- Of course, you need quite a bit of experience and intuition to hit on a good combination of parameters.
- That's why it usually makes sense to do a grid search for hyper-parameter tuning.
- Hyper-parameter tuning with grid search allows us to test different combinations of hyper-parameters and find one with improved accuracy.
- We can use the `h2o.grid()` function to perform a Random Grid Search (RGS). We could also test all possible combinations of parameters with Cartesian Grid or exhaustive search, but RGS is much faster when we have a large number of possible combinations and usually finds sufficiently accurate models.
- Based on the results of each model tested in the grid, we can choose the one with the highest accuracy or best performance for the question on hand

Classification with tree-based models

Decision trees

e.g. Random Forest and gradient boosting trees



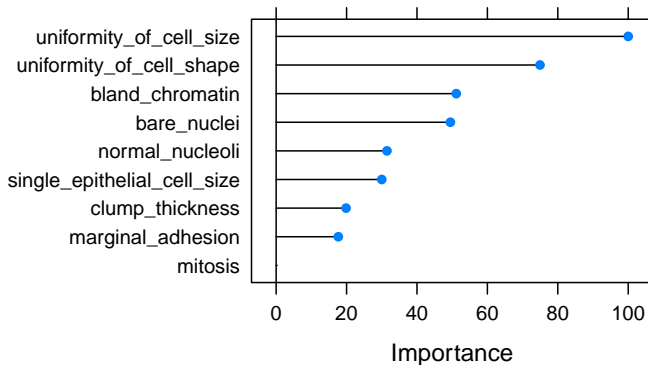
How to build ML models in R

Classification with tree-based models



- We start with a group of samples
 - for each sample, we assign a class: e.g. it comes from a benign or malignant tumor
 - for each sample, we also have a number of features
 - a decision trees separates the data at several nodes to end up with classifications at the final leaves
 - the model learns a conditional structure of discriminative features
-
- Random Forests produces multiple decision trees
 - with some level of randomness
 - Every node in the decision trees is a condition on a single feature, designed to split the dataset into two so that similar response values end up in the same set
 - each tree is evaluated regarding how well it classified the samples (cross-validation)
 - ensemble of all trees is used for prediction
 - better generalization than individual trees

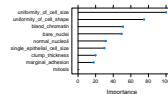
Feature importance



How to build ML models in R

Feature importance

Feature importance



- Not all of the features I created will be equally important to the model.
- A benefit of using ensembles of decision tree methods is that they can automatically provide estimates of feature importance from a trained predictive model.
- Generally, importance provides a score that indicates how useful or valuable each feature was in the construction of the decision trees within the model.
- The more an attribute is used to make key decisions with decision trees, the higher its relative importance.
- Importance is calculated for a single decision tree by the amount that each attribute split point improves the performance measure, weighted by the number of observations the node is responsible for.
- The feature importances are then averaged across all of the decision trees within the model.
- when training a tree, it can be computed how much each feature decreases the weighted impurity in a tree. For a forest the impurity decrease from each feature can be averaged and the features are ranked according to this measure.
- 'information gain' is the measure that tells us how good a tree model is
- the measure based on which the (locally) optimal condition is chosen is called impurity. For classification, it is typically either Gini impurity or information gain/entropy and for regression trees it is variance

Evaluating model performance

**Never use the same data
for evaluation that you used
for training!**

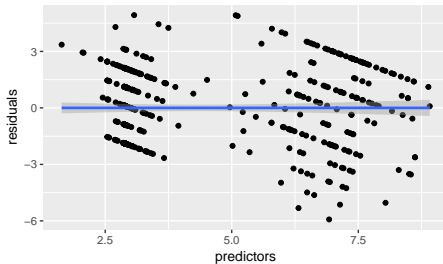
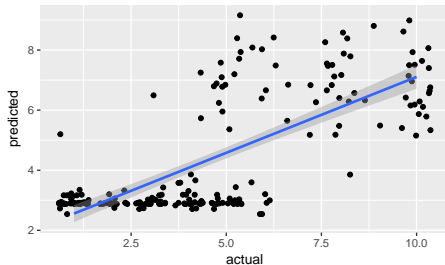
**Never use the same data
for evaluation that you used
for training!**

- Golden Rule of ML: always test model performance on independent data
- otherwise you will get overly optimistic performance measures
- The ultimate performance test for our model will be its prediction accuracy on the test set it hasn't seen before.

Predictions on test data

Regression

- RMSE: 1.97
- R^2 : 0.50



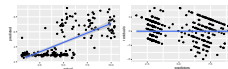
Evaluating model performance

Predictions on test data

Predictions on test data

Regression

■ RMSE: 1.97
 ■ R²: 0.50



- Residuals refer to the difference between the recorded value the predicted value
- Generally, the residuals of a well-fit model should be randomly distributed because good models will account for most phenomena in a data set, except for random error.
- In regression analysis, the term mean squared error refers to the unbiased estimate of error variance:
 - the residual sum of squares divided by the number of degrees of freedom
 - RMSE is a commonly used error metric to measure the performance of regression models.
 - in regression the predictor variable is a real number, therefore to measure the quality of the predicted value from some algorithm you need to find some sort of difference between them. You do this by calculating the square of the error, take the mean across all test objects and take square root
 - A small RMSE means good prediction and large means bad model.
 - RMSE is strongly influenced by outliers!
- R-squared is the ratio of explained variance to the total variance.
 - it is a measure of how much of the variance in y is explained by the model. If R² is close to one, then the model's predictions mirror true outcome

Predictions on test data

Classification

Confusion Matrix and Statistics

##

Reference

Prediction benign malignant

benign 133 2

malignant 4 70

##

Accuracy : 0.9713

95% CI : (0.9386, 0.9894)

No Information Rate : 0.6555

P-Value [Acc > NIR] : <2e-16

##

Kappa : 0.9369

McNemar's Test P-Value : 0.6831

##

Sensitivity : 0.9708

Specificity : 0.9722

Pos Pred Value : 0.9852

Neg Pred Value : 0.9459

Prevalence : 0.6555

Detection Rate : 0.6364

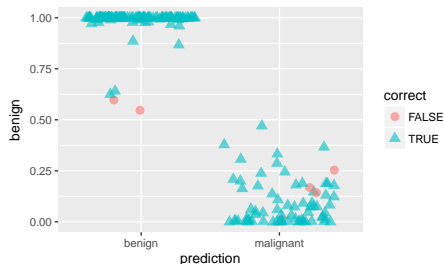
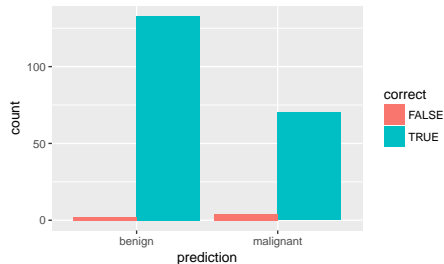
Detection Prevalence : 0.6459

Balanced Accuracy : 0.9715

##

'Positive' Class : benign

##



Evaluating model performance

Predictions on test data

Predictions on test data

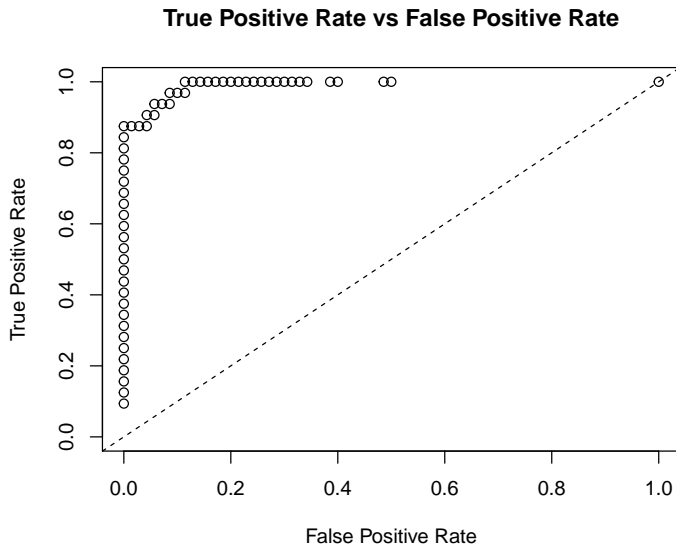
Classification

```
## Confusion Matrix and Statistics
##
##      | 1 0 |
##:--:--:|
## 1 | 10  0 |
## 0 |  0 10 |
##----+--|
## Sum of samples = 20
##
## Accuracy : 1.000000
## No Information Rate : 0.500000
## Kappa : 1.000000
##
## Mcnemar's Test P-Value = 0.000000
##
## Sensitivity : 1.000000
## Specificity : 1.000000
## Positive Predictive Value : 1.000000
## Negative Predictive Value : 1.000000
## Balanced Accuracy : 1.000000
##
## 'Positive' Class = 1
```



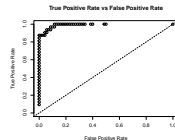
- precision == positive predictive value: fraction of predictions that are correct
- positive and negative predictive values: proportions of true positive and true negative results, depend on prevalence
- prevalence: how often each category occurs in the population
- recall == sensitivity == True Positive Rate: proportion of positives that are correctly identified (e.g., the percentage of people who are correctly identified as having the condition).
- Specificity == True Negative Rate: proportion of negatives that are correctly identified (e.g., the percentage of healthy people who are correctly identified as not having the condition)
- Inverse Precision and Recall: Precision and Recall where positive and negative labels are exchanged
- detection rate == sensitivity
- Accuracy: weighted arithmetic mean of Precision and Recall
- Given a confusion matrix of classification results, the accuracy can be a misleading performance measure. Specifically may falsely suggest above-chance generalizability:
- in binary classification, a training set consisting of different numbers of representatives from either class may result in a classifier that is biased towards the more frequent class
- this classifier may yield an optimistic accuracy estimate
- Balanced accuracy: average accuracy obtained on either class. Based on a confusion matrix
- p-value: one-sided test to see if the accuracy is better than the "no information rate"
- No Information Rate: the proportion of classes that you would guess right if you randomly allocated them
- Kappa: can have values between -1 and 1. 1 or -1 show complete agreement, zero shows complete disagreement
- McNemar's test: statistical test applied to 2 x 2 contingency tables to determine whether the row and column marginal frequencies are equal

Area Under the Curve (AUC)



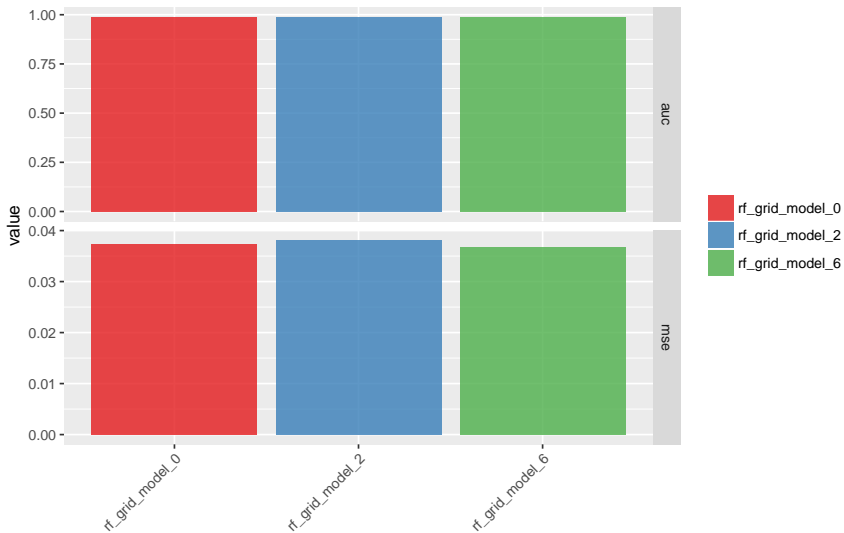
Evaluating model performance

Area Under the Curve (AUC)



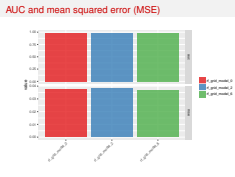
- AUC usually refers to area under ROC curve (mathematically known as definite integral): Receiver Operating Characteristic
- metric for binary classification
- Accuracy deals with ones and zeros, meaning you either got the class label right or you didn't. But many classifiers are able to quantify their uncertainty about the answer by outputting a probability value.
- From a random classifier you can expect as many true positives as false positives. That's the dashed line on the plot.
- A score for a perfect classifier would be 1

AUC and mean squared error (MSE)



Evaluating model performance

AUC and mean squared error (MSE)



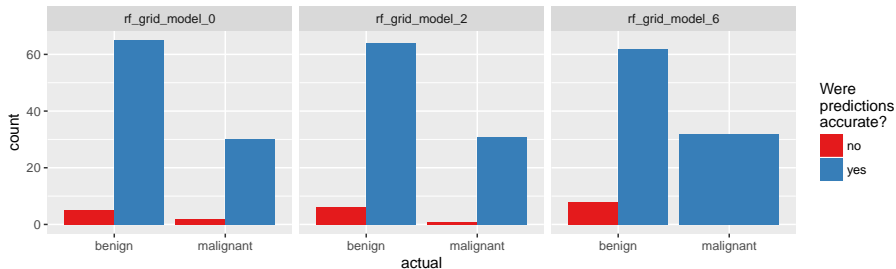
In statistics, the mean squared error (MSE) or mean squared deviation (MSD) of an estimator (of a procedure for estimating an unobserved quantity) measures the average of the squares of the errors or deviations—that is, the difference between the estimator and what is estimated. MSE is a risk function, corresponding to the expected value of the squared error loss or quadratic loss. The difference occurs because of randomness or because the estimator doesn't account for information that could produce a more accurate estimate.[1]

The MSE is a measure of the quality of an estimator—it is always non-negative, and values closer to zero are better. The MSE is the second moment (about the origin) of the error, and thus incorporates both the variance of the estimator and its bias. For an unbiased estimator, the MSE is the variance of the estimator. Like the variance, MSE has the same units of measurement as the square of the quantity being estimated. In an analogy to standard deviation, taking the square root of MSE yields the root-mean-square error or root-mean-square deviation (RMSE or RMSD), which has the same units as the quantity being estimated; for an unbiased estimator, the RMSE is the square root of the variance, known as the standard deviation.

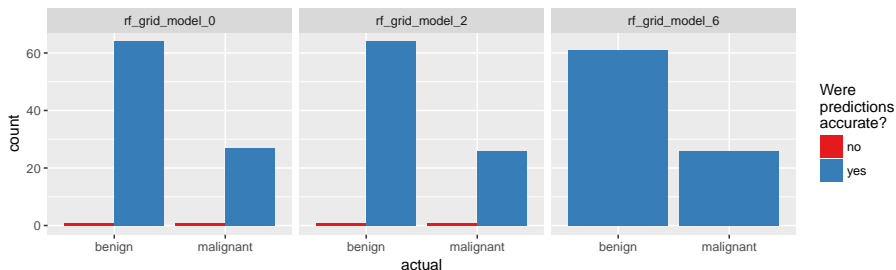
The MSE assesses the quality of an estimator (i.e., a mathematical function mapping a sample of data to a parameter of the population from which the data is sampled) or a predictor (i.e., a function mapping arbitrary inputs to a sample of values of some random variable). Definition of an MSE differs according to whether one is describing an estimator or a predictor.

Predictions on test data

Default predictions



Stringent predictions



Take home messages:

- ...

- 'big data' needs to be big!
- for really meaningful models, data needs to be shared
- the more data, the more accurate and generalizable the models will be
- issues: privacy, platform, quality standards
- ML could make health care more cost-effective by reducing the energy required for interpretation

Thank you for your attention!

Questions?

Slides and code will be available on Github:

https://github.com/ShirinG/Webinar_ML_for_disease

Code will also be on my website: <https://shiring.github.io>

shirin.glander@wwu.de