DATA TECNICA INTERNATIONAL

Supervised Machine Learning Methods

Introduction to algorithms and use cases in neurology and neurogenetics

NIH's Center for Alzheimer's and Related Dementias (CARD)
Data Tecnica International
German Center for Neurodegenerative Diseases
Hampton Leonard
June, 2022

DZNE NIH CARD

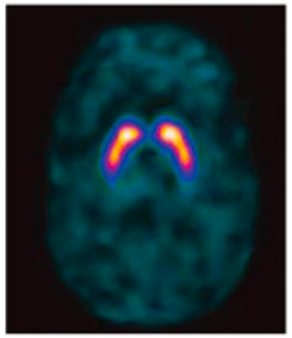
Agenda

1. Intro to supervised machine learning
    a. Data preprocessing
2. Algorithms
    a. Pros
    b. Cons
3. How to assess model performance and choose your model
4. Examples of machine learning applications to neurology
    a. Benefits and limitations

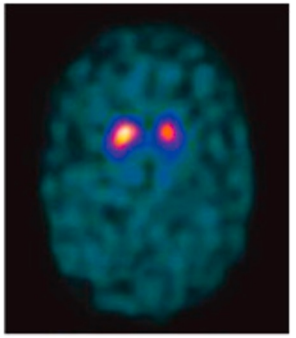
What is supervised machine learning?

- A type of machine learning that is used to predict an outcome variable accurately
  - The input dataset requires pre-defined labels
    - Labels can be discrete or continuous
- Supervised methods will attempt to minimize error through various loss functions
  - Loss functions help the algorithm determine how close they are to modeling the relationships in the data

Supervised ML identifies patterns specific to different labels/outcomes to give us actionable conclusions we may not be able to identify with our human eyes/brain



Healthy Control



Parkinson's Disease

MMSE: 28

MMSE: 22

MMSE: 20

Supervised learning: a few pros and cons

Pros

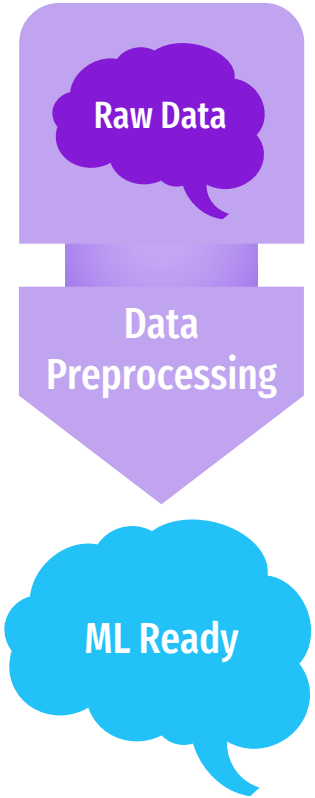
- Can be used for building accurate models for predicting the outcome variable
- Can also be used to find features that have an important relationship to the outcome variable
- Easy to assess accuracy of the models

Cons

- Requires accurate labels (not always possible in the real world)
- Prone to being ‘overfit’ - too specific to the training dataset
- Typically you need good representation for all classes, unbalanced data can produce misleading results

Before we start...

- Prepare your data
  - Algorithms often get all the credit, but data preprocessing is just as important
- Get to know your data
  - This could mean looking at how much data is missing, how many variables are correlated, distribution of outcome variables etc.
- Scaling and dimensionality reduction
  - 2 important techniques to get data ready for an ML algorithm and improve performance



Scaling

Standardization

Makes variables have a mean of 0 and a standard deviation of 1 within that variable

Best for when data has a somewhat Gaussian (normal) distribution

Outliers are not constrained by upper or lower bounds

Normalization

Forces variables within the boundaries of 0 to 1, 0 to the smallest value of a variable, 1 to the largest

Scaled proportionally between 0 and 1, underlying distribution does not matter

Outliers are constrained within bounds

Why scale before ML?

01

In many cases, the features we want to include in our models are on different scales

02

An unimportant feature may be on the scale of [500-800] while an important feature may be on the scale of [5-10]

03

In some algorithms (like distance based), the unimportant feature would count more

04

Many algorithms will perform poorly on unscaled data - this includes dimensionality reduction

05

Scaling improves computation time for some algorithms

Dimensionality reduction

- Big data is exciting but poses a lot of challenges
  - It is not always possible to use every piece of data available because it can be expensive computationally and inconvenient

Principal Component Analysis

Creates new features that are linear combinations of the original features

New features are made to maximize variance in the data as well as minimize redundancy

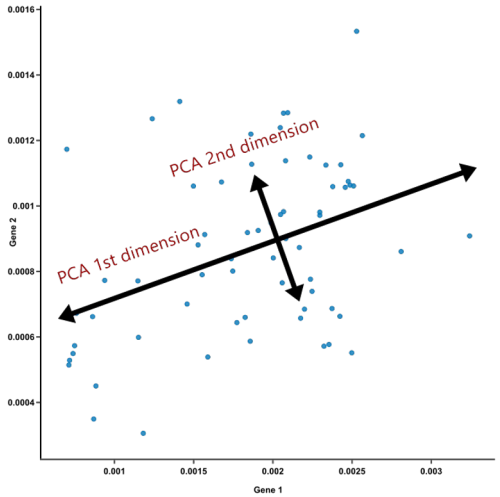
UMAP

Same reduction goal as PCA but can preserve some of the structure of the data

Calculates data points that are close together and then recreates the same structure in a smaller number of dimensions

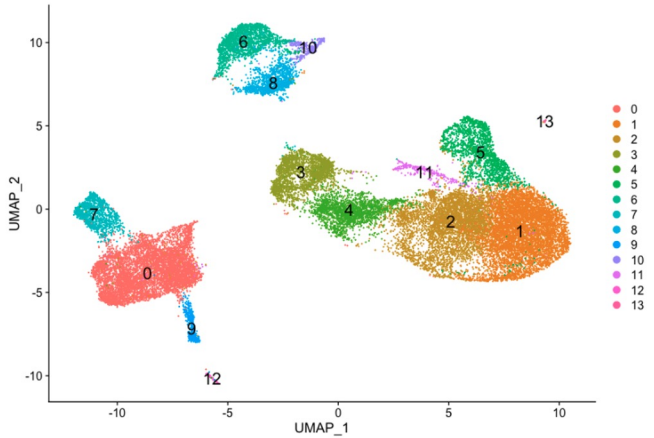
Dimensionality reduction

Principal Component Analysis



- PCA is great for tasks that are relatively easy or time-sensitive
- If interpretation of reduction is needed, PCA is typically more interpretable than UMAP

UMAP

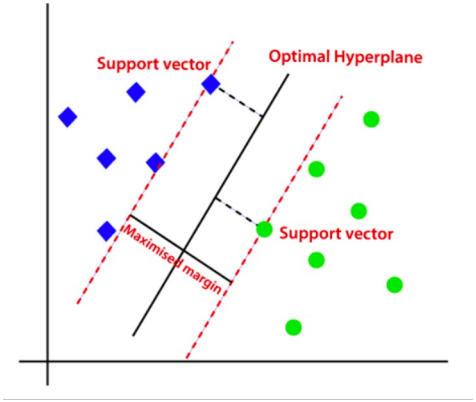


- UMAP requires more computational time and memory, but it preserves some local and global structure
- This makes it great to use with clustering algorithms

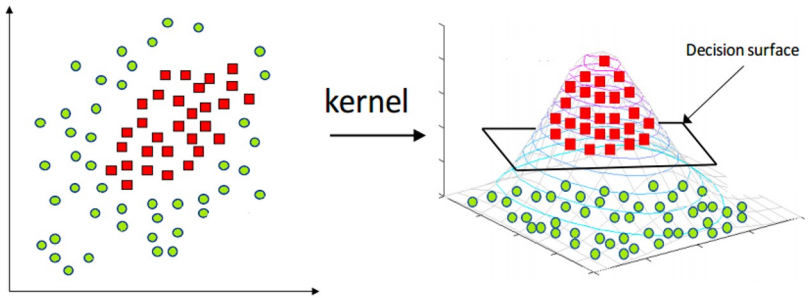
Support vector machine (SVM)

- SVM attempts to obtain the maximal separation of data using a line or hyperplane
- Support vectors are points that are close to the hyperplane that help determine the optimal location
- Each feature is weighted
    - The higher the value, the more important it is to determine the position of the hyperplane
- Linear functions are the most straightforward, but not all data is linearly separable
    - In this case, SVM has different ‘kernel functions’ that can transform the data to a higher dimension

Linear SVM



Non-linear SVM



Pros and Cons

Pros

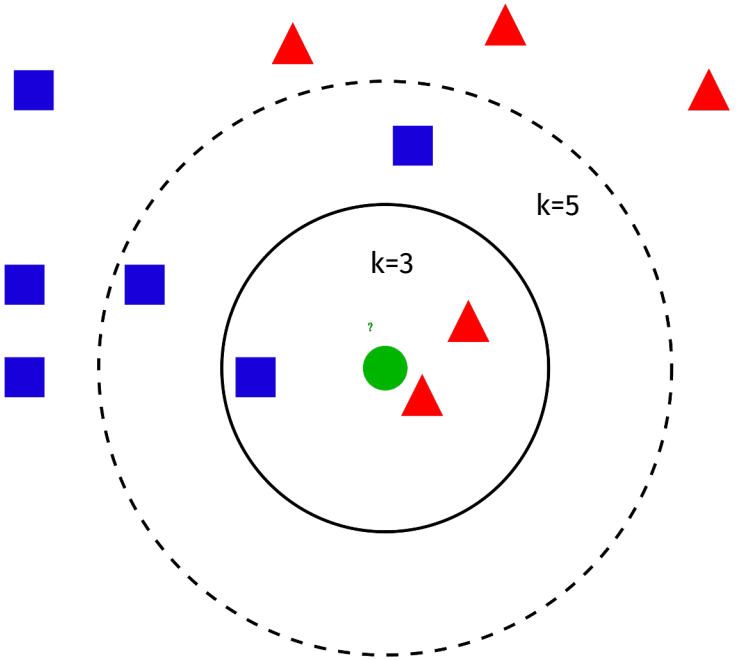
- Works very well on high-dimensional data
- Also works well in cases where there are many more features than samples
- Outliers have less impact

Cons

- Performs poorly if there is high class overlap (classes are not separable)
- Typically slow to train
- Not highly interpretable in high dimensional spaces

k-nearest neighbor (kNN)

- kNN attempts to classify new data by using the proximity of the data closest to that new data
- You choose the value of k
  - k=1: new points will be predicted to have the same outcome as the point closest to it
  - k=5: new points will be predicted to have the same outcome as the majority outcome from the 5 closest points around it
- Proximity of data is determined by distance functions that you can specify



Pros and Cons

Pros

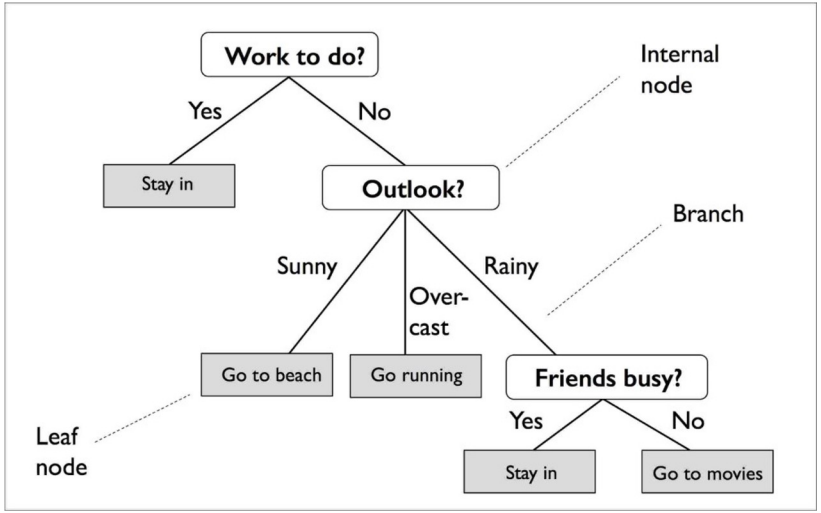
- Easy to implement
- Works well when you have data with multiple classes
- Good for nonlinear data

Cons

- Sensitive to outliers
- Does not handle missing data well
- Does not handle high-dimensional data well
- Does not do well with imbalanced classes
- Slow with large datasets

Decision trees

- Decision trees attempt to identify sets of features in which that set of features only correspond to one class or label
- Trees consist of nodes, branches, and leaves
    - The root node is the top decision node, it is the attribute that best classifies the data
    - Branches represent decisions or rules
    - Leaf nodes represent the outcome decision
- Trees can use different methods to make decision splits and will grow until each class is represented in it's own leaf node or until another threshold is met



Pros and Cons

Pros

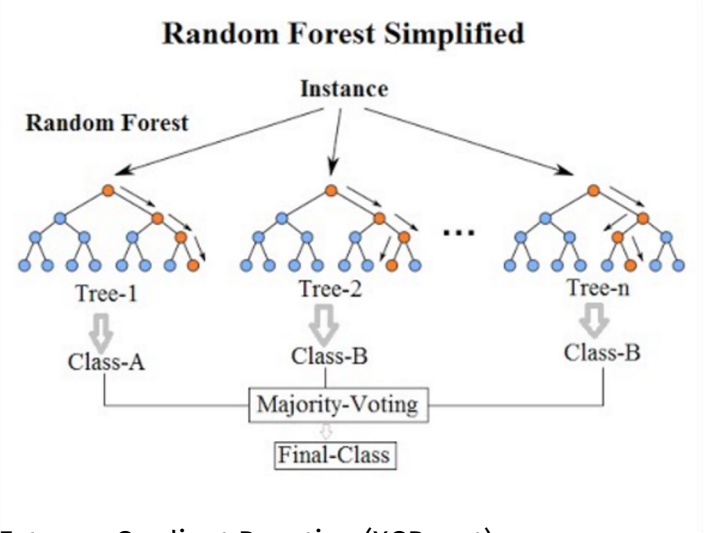
- Interpretable and tree decision making methods make it very easy to determine important features
- Very good at handling missing data
- Doesn’t require data to be scaled beforehand
- Can be used for feature selection for other models

Cons

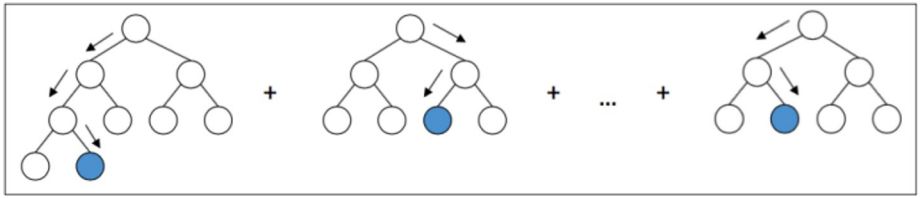
- Prone to overfitting
- Relatively unstable - small changes in the training dataset may result in a completely different tree structure

Bonus: random forest and extreme gradient boosted trees

- Random forest is a collection of decision trees
  - Each tree is trained on a random subset of the data
  - A voting system is used to determine class membership/probability
  - This reduces the overfitting problems with using single decision trees
- Boosting is also a collection of decision trees, but instead of many trees fit on different subsets of the data, boosting fits one tree at a time where each tree learns from the previous one
  - Fit first tree to the whole data set, the second tree is fit to the residual error of the first, the third is fit to the residual error of the second, and so on

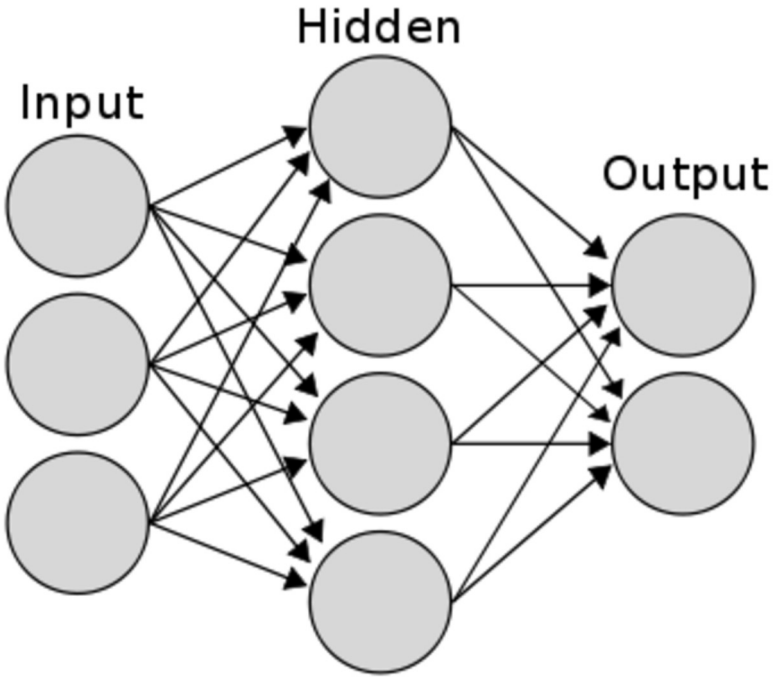


Extreme Gradient Boosting (XGBoost)



Neural networks

- Networks are organized into layers with interconnected nodes
  - The input layer is the data
  - The first hidden layer is the combinations of all input features
    - Each combination is given a different weight depending on the importance of the features
  - The output layer corresponds to class prediction
- Relies on the concept of ‘backpropagation’
  - Once a class probability has been assigned, the network checks how well it did and will go back to change the weights of each feature combination until the result is a (hopefully) accurate and generalizable model



Pros and Cons

Pros

- Many applications! Classifying tumors from medical images, identifying rare cell populations from blood samples, predicting disease outcomes, etc.
- Can learn and model non-linear and complex relationships

Cons

- Not as interpretable
- Can be more difficult to implement compared to simpler algorithms
- Can be computationally expensive

Model Assessment

Why do we need to evaluate our models?

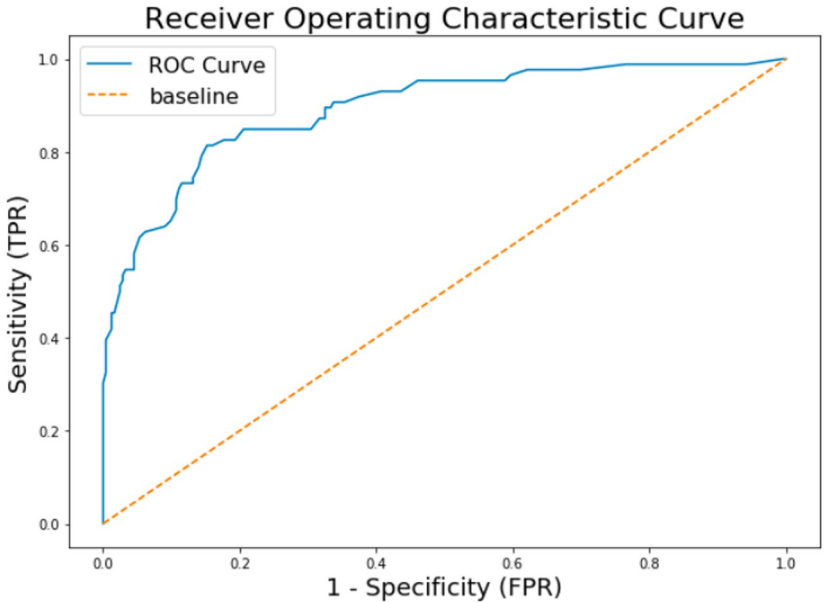
We need to be able to estimate performance on unseen data - otherwise we will not know if the model is generalizable

We want to identify the best model for the problem

If the model isn't accurate/doesn't fit the data well, we can't say much about interpretation of the model or feature importance

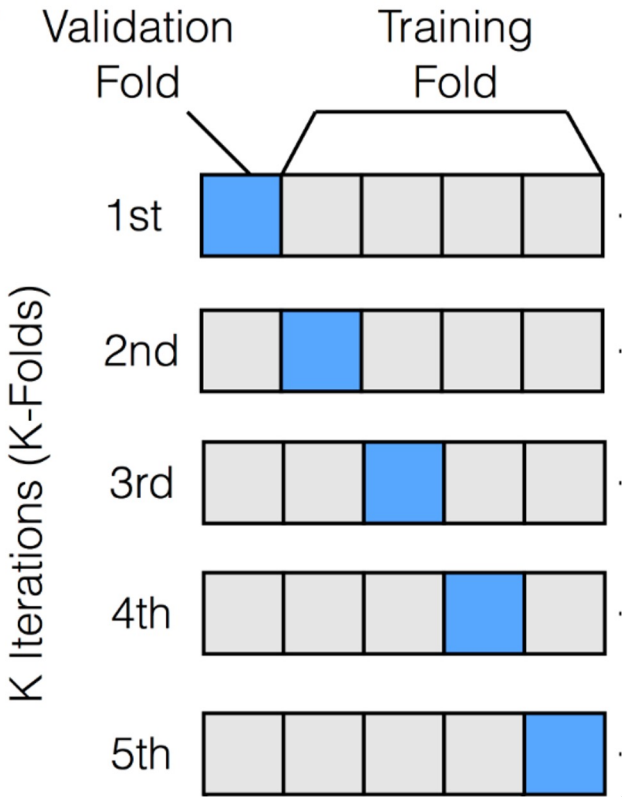
R-squared and ROC/AUC

- R-squared estimates how well your model is fitting the data
    - Estimates how much variance in the data is explained by the model (best to use adjusted R-squared)
    - Commonly used with linear regression but can be extrapolated to other models
    - Mean squared error is another metric useful to estimating model accuracy on a continuous outcome
- Receiver operator curve and the area under it is used to determine accuracy of classification models
    - ROC represents the trade-off between sensitivity and specificity and AUC represents the degree of separability between classes
    - AUC of 1 is perfect, 0.5 is random guessing



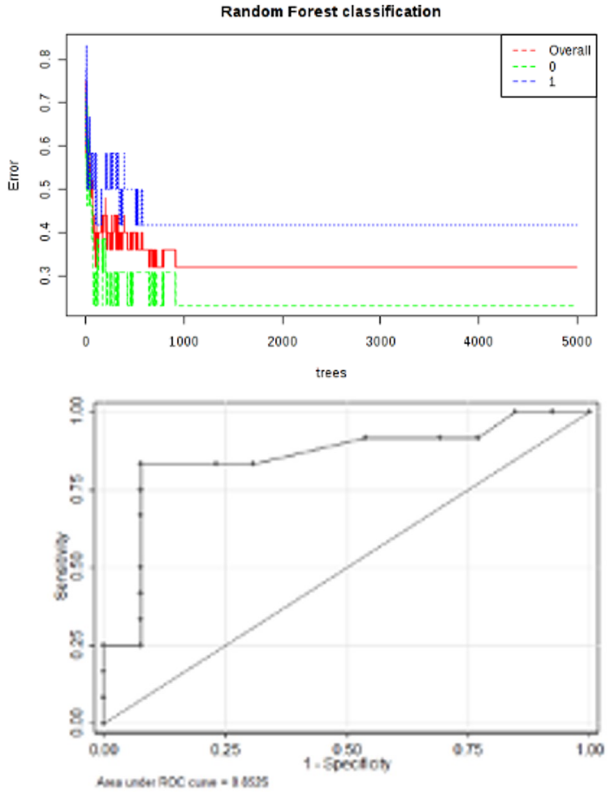
K-fold cross validation

- Best option for validation would be to test on a completely new dataset not used in training, but this is not always possible
- K-fold cross validation is a resampling method that tests the model on k folds of the dataset
    - Shuffle, split dataset into k groups
    - Take 1 group of k and hold out, train model on remaining k-1 groups
    - Evaluate model on held out group, reiterate k times
- Allows us to estimate generally how our model will perform on unseen data
    - A more robust option than splitting off a test set since you are technically testing your model on an entire ‘unseen’ dataset!



Application of ML #1

- *Metabolome-based signature of disease pathology in MS*, Andersen et al. 2019
- Assessed 400 plasma metabolites in 12 individuals with MS and 13 healthy controls
- Used the random forest algorithm to determine important metabolites in separating MS from control
    - 6 metabolites nominated and reached an AUC > 0.8 for prediction of MS



Benefits and limitations

Benefits

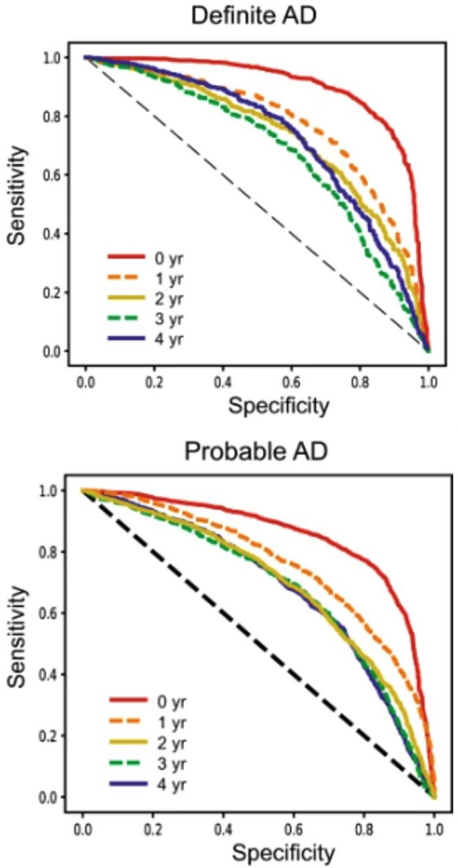
- Random forest is able to provide information about feature importance
- Can potentially give us some ideas of what to investigate further in terms of dissecting the etiology of MS or for developing additional biomarkers

Limitations

- Sample size small
- Very likely to be overfit
- No replication
- Data very homogenous (only males in the dataset)

Application of ML #2

- *Machine learning prediction of incidence of Alzheimer's disease using large-scale administrative health data*, Park et al. 2020
- Assessed model prediction of future incidence of AD with 40,736 elders from 4,894 unique clinical features including ICD-10 codes, medication codes, laboratory values, history of personal and family illness and socio-demographics
- Compared SVM, logistic regression, and random forest to predict 'definite AD' and 'probable AD'
    - RF showed the best performance with accuracy 0.823 and AUC of 0.898, accuracy decreased when attempting to predict earlier



Benefits and limitations

Benefits

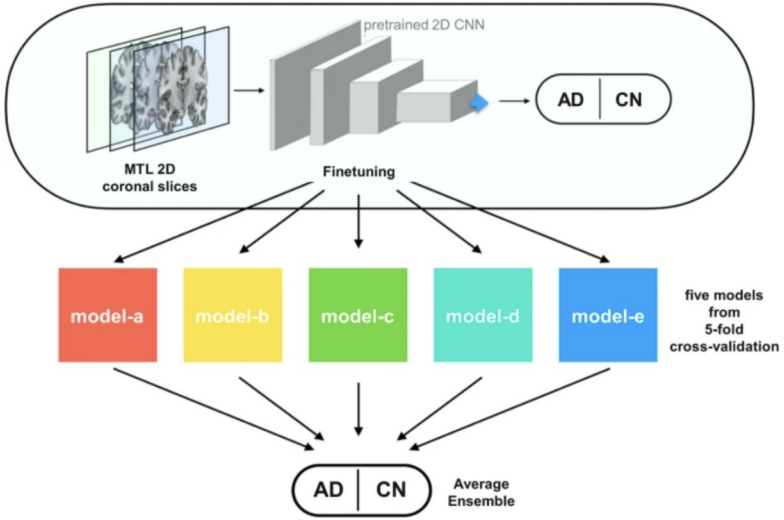
- Using large-scale administrative health data provides large sample size and utility
- Random forest, SVM, and logistic regression all interpretable algorithms
    - hemoglobin level determined as most important feature for prediction

Limitations

- Best accuracy for predicting ‘definite’ AD 4 years away from diagnosis at 0.66... clinically useful?
- Messy labels (AD not always accurately diagnosed)

Application of ML #3

- *Identification of Alzheimer's disease using a convolutional neural network model based on T1-weighted magnetic resonance imaging*, Bae et al. 2020
- Assessed model prediction of AD vs control using 2D slices of T1-weighted MRI images
- Used a convolutional neural network
    - Trained and tested two different ancestries, East Asian and European
    - AUC of 0.88 in validation across populations and 0.91 in validation within the same population



Benefits and limitations

Benefits

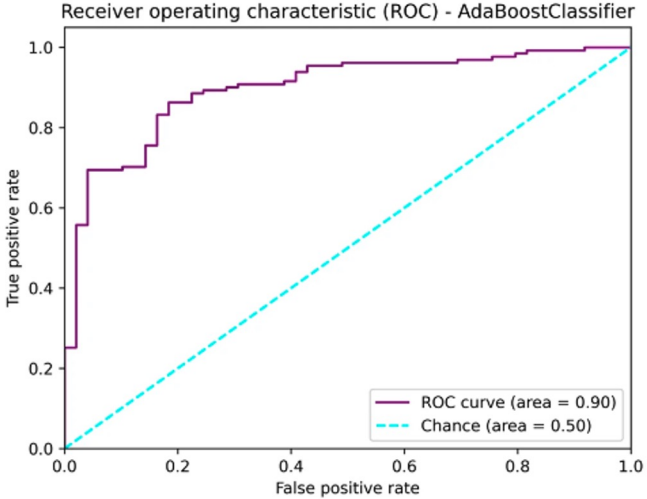
- CNNs are very effective at learning from images, 2D images have lower computational complexity
- Testing models on other populations is necessary for generalizable use
- MRI(and imaging in general)-based models may be at an advantage for improving diagnosis

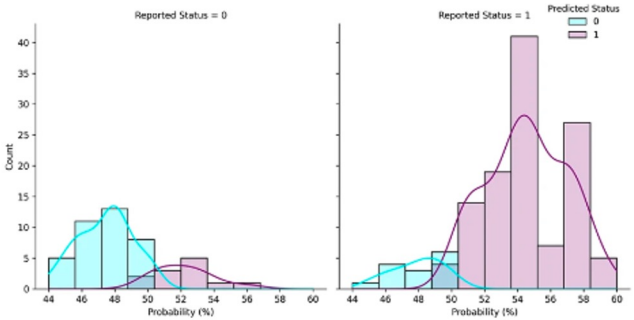
Limitations

- Low interpretability... but interpretability is not necessarily the goal
- Imaging data is messy - different scanners, different MRI protocols etc.
- As with all supervised machine learning, the problem of human error is assigning labels persists

GenoML

- Automated ML for genomics data
- Prepares data for ML with a number of options including getting rid of collinearity, feature selection, imputation, scaling, and pruning of variants (if you are including genetic data)
- Tests a wide range of algorithms and provides various model assessment metrics
- Options to tune and test your model
- Open source and under development! https://github.com/GenoML/genoml2





Makarious et al.

Thank you!