\section{Machine Learning}

\subsection{Models}

\paragraph{Weighted Linear Regression}:

Linear regression is easy to understand and interpret; the sign and absolute value of the coefficient indicates its relationship to the outcome. If the number of coefficients is reasonable, an observer can quickly understand how the model behaves.

While linear regression is ill-suited to most real-world datasets because of their complex decision-boundaries, if the decision boundary domain is separated into smaller domains which are linearly separable, it can be used effectively on each subdomain.

These two points make it a very interesting tool for interpretation. A major drawback and important point is that this model is subject to interactions between variables.

Formula:

$y\_{i}=\beta \_{0}+\beta \_{1}x\_{i1}+ ... +\beta \_{p}x\_{ip}+\epsilon \_{i}, i=1,... ,n,$

\paragraph{Logistic Regression}

Logistic regression is extensively used in the medical field. Once again, the coefficients can be easily understood, as a log odds relation to the outcome. While in theory it does not detect complex patterns, it has proven to perform well on various datasets **[source]**. Its extensive use in the medical field and ease of interpretation make it a good comparative point to other models.

This model is based on the linear regression model, with the use of a sigmoid function, which allows to transform the regression into a binary classification problem. Like most models, modifications can be made so as to extend the applicability of this structure.

Sigmoid Function:

$S(x)=\frac{1}{1+e^{-x}}$

\paragraph{Naive Bayes Bernoulli Classifier}

Once again, this model is extensively used and is fast to train. There is a strong assumption in these models that features are independent from each other which is often not the case especially in real-world data. Moreover, if features are highly correlated, the classification becomes more difficult.

While it is a good starting point and its results are interpretable, it is usually not well suited to complex datasets. [**Source, and also include why we re using it … it is in the top 3 most used models from that study I think, with svm and knn or lr something like that]**

The Naive Bayes models are a class of models which base their choice on prior probabilities. Each feature's distribution is modeled as a Normal distribution. The Bernoulli variant of this model is made for binary inputs, and features are assumed to follow a Bernoulli distribution. **[following: ]** *notes for me, should remove probably from the thesis – too textbook like] The posterior probability is computed by multiplying the likelihood by the prior probability. The class yielding the highest posterior probability will be assigned to that data point. To evaluate the posterior, only consider a fraction of the data points in the likelihoods.*

Formula: conditional probability:

Posterior = prior\*likelihood/evidence

Posterior: $\hat{y} = arg\max\limits\_{k\in \{1,\cdots ,K\}} p(C\_{k}) \sum\_{i=1}^{n}p(x\_{i} \mid C\_{k})$

Bernoulli:

$p(x \mid C\_{k})=\prod \_{i=1}^{n}p\_{ki}^{x\_{i}}(1-p\_{ki})^{(1-x\_{i})}$

\paragraph{Support Vector Machine}

Support vector machines can have different kernels used if the boundary decision are not linear. They are great for large datasets **[source: I think for large and sparse datasets ?].**

**Include formula / image**

\paragraph{Random Forest}

Random forest algorithms are interesting in the medical field, as they apply a rule-based algorithm, which is similar to how clinicians can make a diagnosis. Moreover, they have been shown to perform well on sparse and tabular data [**source]**. They are computationally expensive, but while they tend to overfit, measures to fight this overfit are very well documented (increase number of trees / reduce depth of tree). This algorithm usually requires some sort of hyper-parameter tuning to be performant. Moreover, it requires data pre-processing and can be biased by high cardinality features. This makes it an interesting model to compare with and without tuning / pre-processing.

The random forest algorithm is a ensemble of decision trees. Decision trees use decision rules at successive nodes in order to separate the dataset and obtain homogeneous final nodes. A random forest is a group of such decision trees, each trained independently (and usually with a reduced number of data points and data features). The overall model prediction is the majority vote of the ensemble of decision trees.

**Include formula / image**

\paragraph{Neural Network / Multi Layer Perceptron}

MLPs usually require extensive tuning and pre-processing. While they are not particularly well suited to tabular data compared to other algorithms, they can detect complex patterns. Another drawback is that they usually require a lot of data points to be trained efficiently. Using this model architecture can highlight the need for hyper-parameter tuning.

**[below: textbook like so make it into 1 sentence]** The architecture is defined by one input layer, one or more hidden layer, and an output layer. The connection schemata from one layer to the next can vary. The model output is then compared to the label, and the error (according to a set loss function) is back-propagated through the layers, which makes it so connections are strengthened or weakened. At each node, there is an activation function (which is non linear) so that the overall result of the model is not a great linear combination but contains many nonlinearities - this allows the model to capture complex behaviors in the data. As the output goes successively through multiple nodes, the input is further and further processed until the final layer where it is assigned an outcome. This architecture in theory can detect incredibly complex patterns. However it is computationally expensive, and requires hyper-parameter tuning.

\subsection{Evaluating Model Performance}

Concerning the results, there are also very direct ways to interpret them. It is important to remember that the results need to be analyzed before delving into their interpretation and the way the model predicted them. There are several metrics that are particularly of use in the medical field.

While accuracy is a good metric, it can be easily fooled in cases (for example) of imbalanced labels. Balanced accuracy, precision and recall are metrics that are commonly used in the field of machine learning to yield a more informative result. However, in the medicine field, the metrics that are preferred are sensitivity, specificity and AUC score. Sensitivity and specificity once again are good indicators, however they depend on the threshold at which they are considered – AUC score however does not, and therefore gives a more complete indication of the model performance overall – instead of at a considered decision threshold.

**[source : logic progression from accuracy to AUC; pitfalls, understanding their use; also add in statistical relevance}**

Moreover, it is important to remember that the size of the dataset greatly affects metrics, which is why effective size **[source and is effective size the actual name?]** should be considered. Once again, in the medical field, special importance should be brought to metrics that do not depend on the disease prevalence. In the case of the data used here, this is of paramount importance because while the disease prevalence could be known in a certain country, it is harder to evaluate it in a population, and as discussed beforehand, the data set used might not be indicative of the true population distribution.

You can evaluate these and tune your model to maximize them, using a validation set. The final metrics should however be evaluated on a hold-out test set.

Some notations: TP (True Positive), TN (True Negative), FP (False Positive), FN (False Negative).

The metrics that will be used are: **[should maybe rephrase the definition to be a measure that is specific to the medicine / diagnosis context]**

\begin{itemize}

\item Accuracy: Quantifies the overall effectiveness of the model in distinguishing positives and negatives: $\frac{TP+TN}{TP+FP+TN+FN}$

\item Precision: quantifies how many positive diagnosis the model wrongly predicted: $\frac{TP}{TP+FP}$

\item Recall (Sensitivity): quantifies how many positive diagnosis the model missed: $\frac{TP}{TP+FN}$

\item Specificity: quantifies how many negative diagnosis the model missed: $\frac{TN}{TN+FP}$

\item True Positive Rate (TPR): quantifies the rate of true positives with regards to the entire positive population: $\frac{TP}{TP+FN}$

\item False Positive Rate (FPR): quantifies the rate of true negatives with regards to the entire negative population: $\frac{FP}{FP+TN}$

\item Precision-Recall curve: Allows to visualize the trade off between precision and recall (sensitivity).

\item Receiver Operating Characteristic (ROC) Curve: plots the true positive rate vs. false positive rate which amounts to plotting recall vs. (1-specificity)

\item Area Under the Curve (AUC) is a one-number (area) summary of the ROC curve.

\end{itemize}