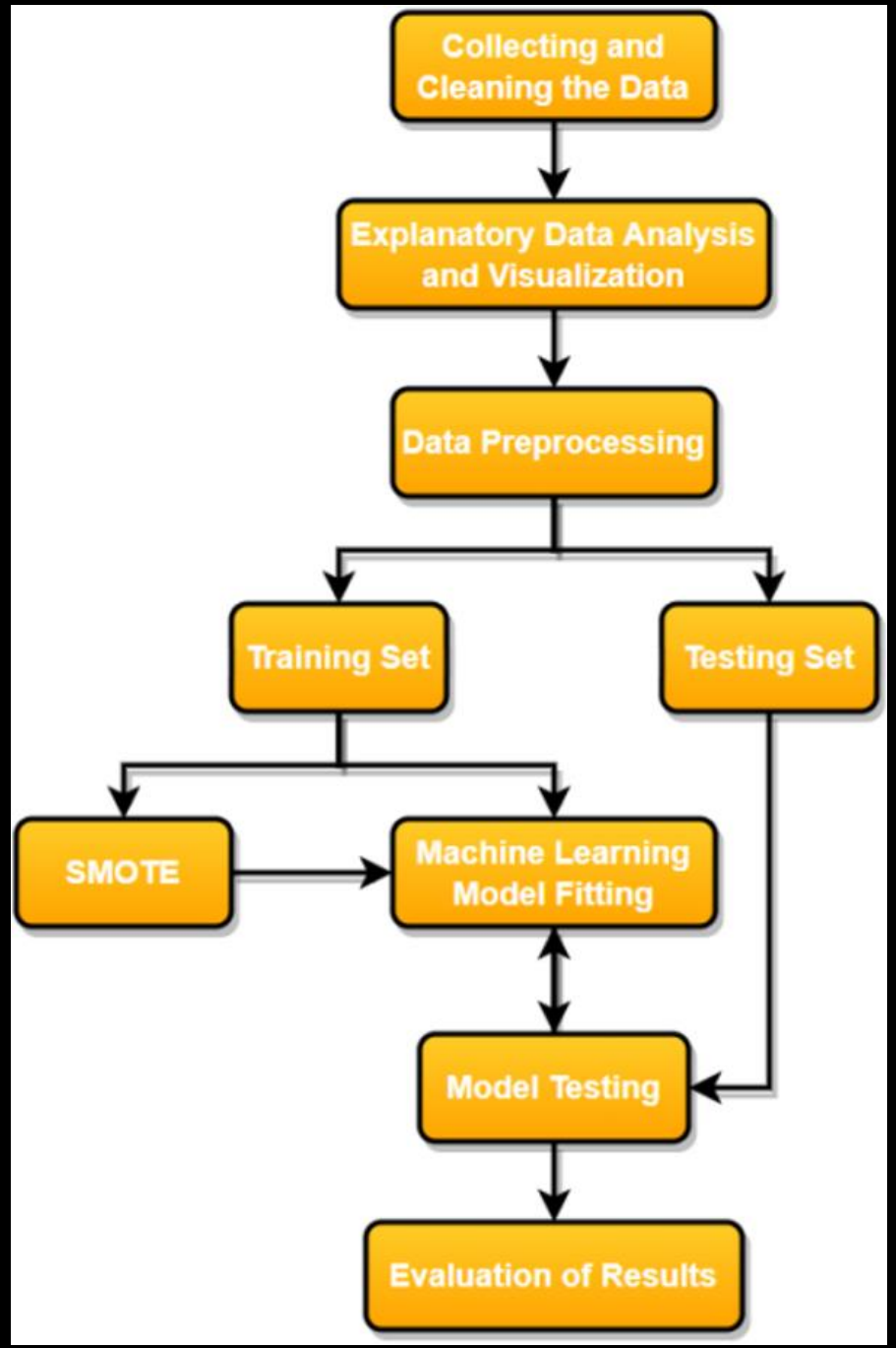
Predicting Cirrhosis of the Liver Using Machine Learning

Dan Barstow
B.S. Mathematics: Operations Research
barstow2@illinois.edu

Outline

- **Goals**
  - Successfully classify cirrhosis using machine learning techniques on a dataset from a clinical trial.
  - Determine which features contribute most to the predictions and see which model performs best.
  - Incorporate suggestions from other researchers.
- **Suggestions**
  - Imputation with the mean for dealing with missing features in samples.
    - Pros: Simple, Doesn't affect the overall mean of the sample data.
    - Cons: Introduces some bias, reduces the variance of the imputed data.
  - Use an SVM with hyperparameter tuning.
    - Pros: Can achieve very high accuracy.
    - Cons: Long training time, Prone to overfitting.
  - SMOTE (Synthetic Minority Oversampling TEchnique) to balance classes.
    - Pros: Doesn't require more data collection, Can increase generalization potential.
    - Cons: Can introduce bias, leading to overgeneralization.



The Dataset

- From a Mayo Clinic trial of D-penicillamine as an alternative for liver transplantation (1974-1984, originally unpublished).
- Dickson et al. create the Mayo Model to predict survival rates of patients with advanced liver disease (1989).
- Markus et al. use the Mayo Model to simulate a control group to assess the efficacy of liver transplantation and the predicted survival rates among patients in the original study who ended up getting a transplant (1989).
- Data made publicly available in *Counting Processes and Survival Analysis* - Fleming & Harrington (1991).
- Data posted to Kaggle (2021).

Out[4]:

	ID	N_Days	Status	Drug	Age	Sex	Ascites	Hepatomegaly	Spiders	Edema	Bilirubin	Cholesterol	Albumin	Copper	Alk_Phos	SGOT	Tryglicerides	Platelets
0	1	400	D	D-penicillamine	21464	F	Y	Y	Y	Y	14.5	261.0	2.60	156.0	1718.0	137.95	172.0	190.0
1	2	4500	C	D-penicillamine	20617	F	N	Y	Y	N	1.1	302.0	4.14	54.0	7394.8	113.52	88.0	221.0
2	3	1012	D	D-penicillamine	25594	M	N	N	N	S	1.4	176.0	3.48	210.0	516.0	96.10	55.0	151.0
3	4	1925	D	D-penicillamine	19994	F	N	Y	Y	S	1.8	244.0	2.54	64.0	6121.8	60.63	92.0	183.0
4	5	1504	CL	Placebo	13918	F	N	Y	Y	N	3.4	279.0	3.53	143.0	671.0	113.15	72.0	136.0

Description of the features in each sample

- `ID` : The case number.
- `N_Days` : The number of days between registration and the earlier of death, liver transplantation, or study analysis time in July, 1986.
- `Status` : Whether the patient was still alive at the time of the statistical analysis in 1986.
- `Drug` : Whether the patient was given the drug D-penicillamine or a placebo in the trial.
- `Age` : The patient's age in days.
- `Sex` : Whether the patient is male or female.
- `Ascites` : Presence of ascites which is a build up of fluid in the abdomen.
- `Hepatomegaly` : Presence of an enlarged liver.
- `Spiders` : Presence of dilated blood vessels which have a spider-like appearance.
- `Edema` : Presence of tissue swelling (typically in the feet or ankles).
- `Bilirubin` : Amount of Bilirubin present in blood measured in mg/dl (breaks down heme from old red blood cells).
- `Cholesterol` : Amount of cholesterol present in blood measured in mg/dl (provides structure to cell membranes).
- `Albumin` : Amount of albumin present in blood measured in mg/dl (keeps fluid from leaking into bloodstream).
- `Copper` : Amount of copper present in urine measured in μg/day (regulates nerve cells and immune system).
- `Alk_Phos` : Amount of Alkaline phosphatase present in blood measured in IU/L (an enzyme which can leak into the bloodstream and cause issues).
- `SGOT` : Amount of SGOT (also called AST) present in blood measured in IU/L (another enzyme which can leak into the bloodstream and cause damage).
- `Triglycerides` : Amount of triglycerides in blood measured in mg/dl (a type of fat which if too high can become an issue).
- `Platelets` : Number of platelets per-cubic-millimeter of blood / 1000 (helps blood to clot).
- `Prothrombin` : Time in seconds required for blood clots to begin forming.
- `Stage` : Stage of liver disease (stage 4 is Cirrhosis).

Cleaning the Data

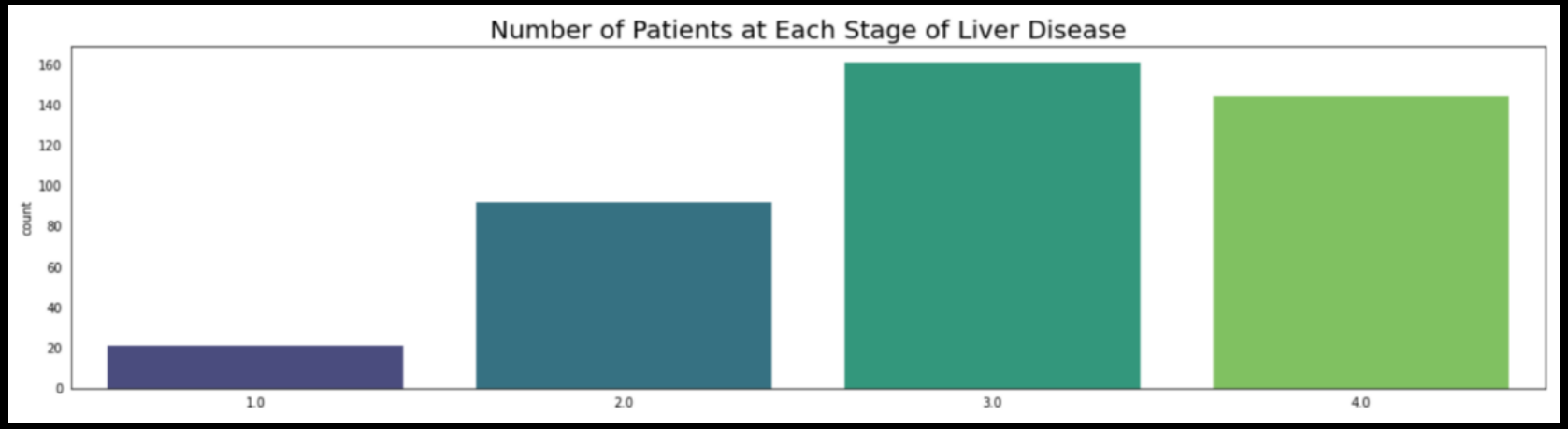
- In the original study, 312 out of the total 418 participated in the trial, and 112 only consented to have some measurements taken.
- `Age` was changed from days to years.
- The missing categorical features (Ascites, Hepatomegaly, etc.) were imputed with the most frequent classification.
- The missing numerical features (Cholesterol, Triglycerides, etc.) were imputed with the median.
- This has the advantage of not changing the actual mean in the data at a cost of introducing some bias by reducing the variance.

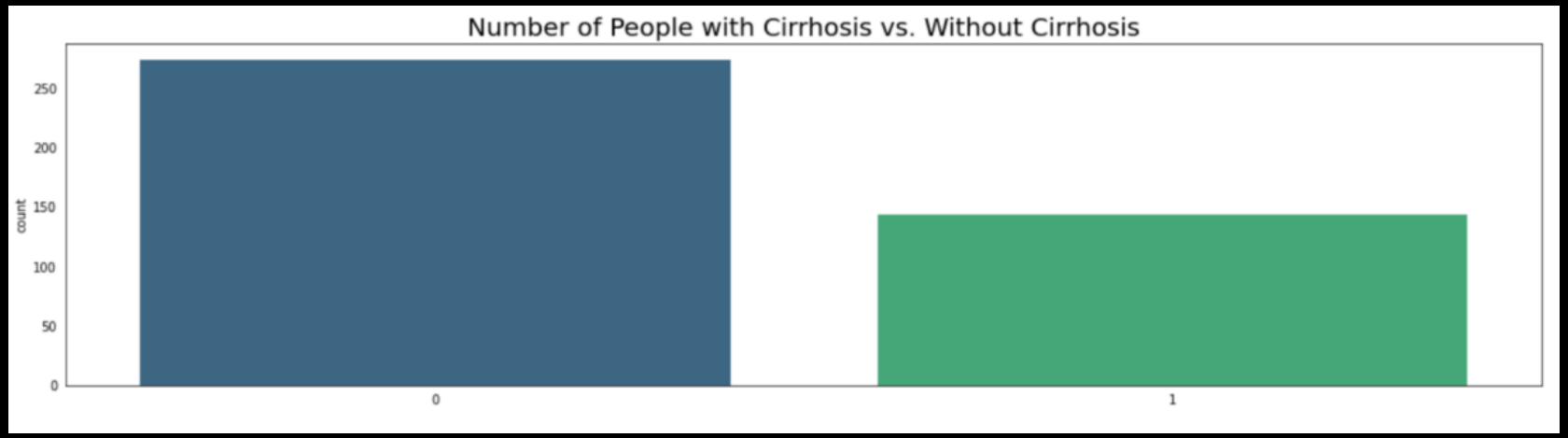
```
In [10]: data.isna().sum()
```

```
Out[10]:
Age              0
Sex              0
Ascites        106
Hepatomegaly   106
Spiders        106
Edema            0
Bilirubin        0
Cholesterol    134
Albumin          0
Copper         108
Alk_Phos       106
SGOT           106
Trygliceride   136
Platelets       11
Prothrombin      2
```

Cleaning the Data

- The `Stage` variable was reclassified as binary.
- We see an imbalance in the target class.
  - 2/3 no cirrhosis
  - 1/3 cirrhosis
- Some models are sensitive to this imbalance and could make worse predictions.
  - SMOTE attempts to fix this.





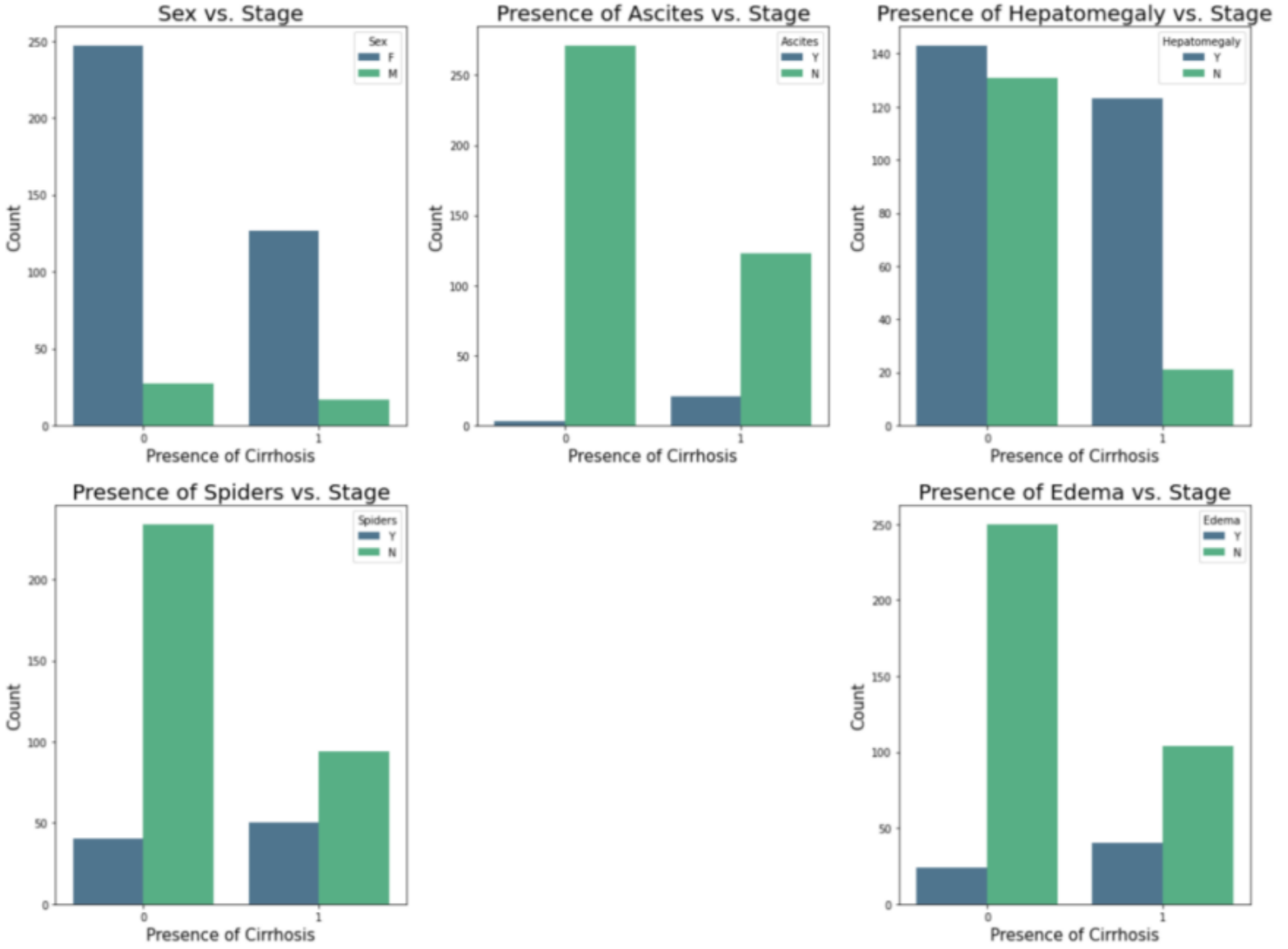


Figure 8: Bar plots of the binary features vs. the target 'Stage'.

Explanatory Data Analysis

Categorical Features

- We see that Presence of `Ascites`, `Spiders`, and `Edema` are slightly more common in patients with cirrhosis.
- We also see few patients with cirrhosis don't have `Hepatomegaly`.
- `Sex` appears to have little relationship with cirrhosis.

Explanatory Data Analysis

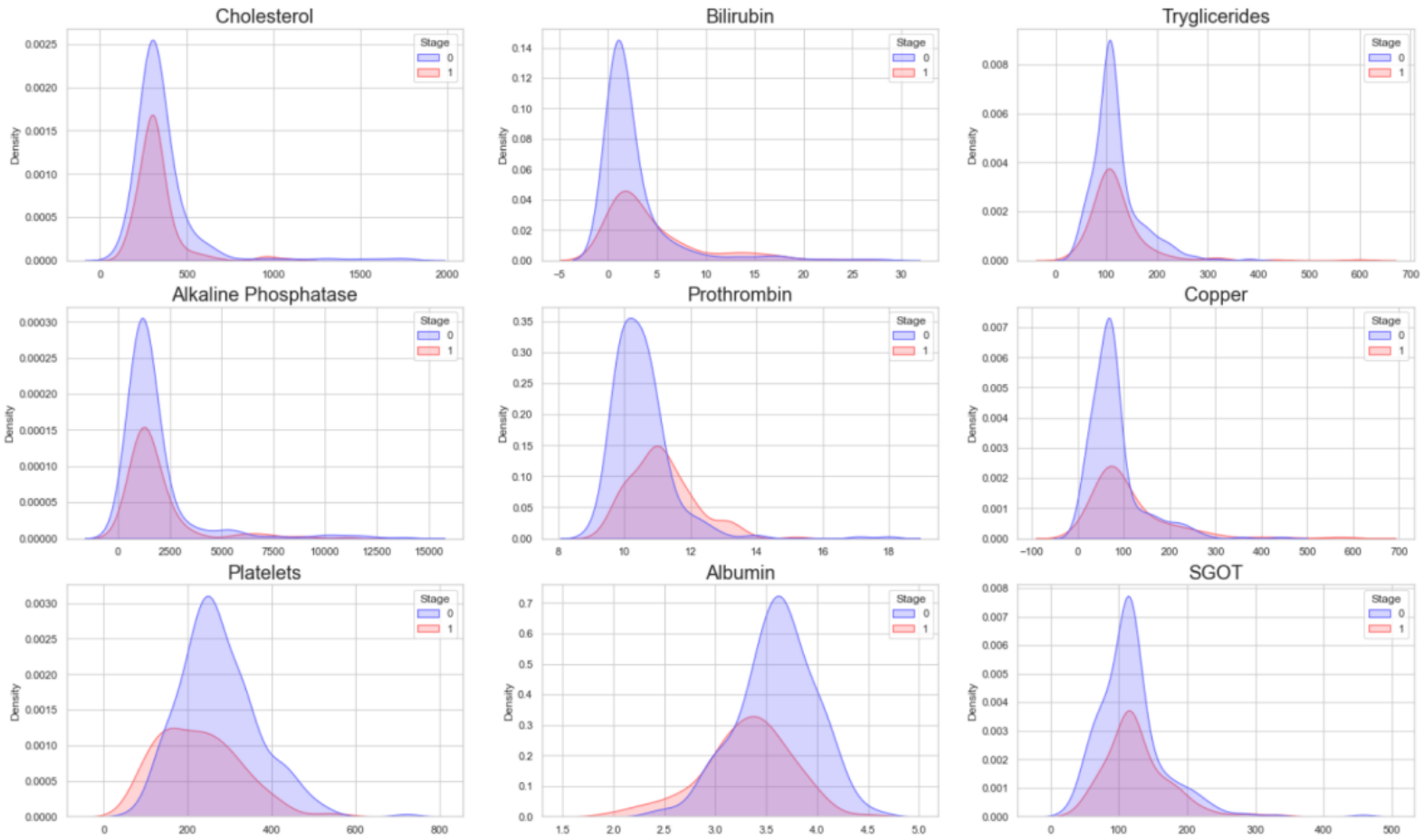
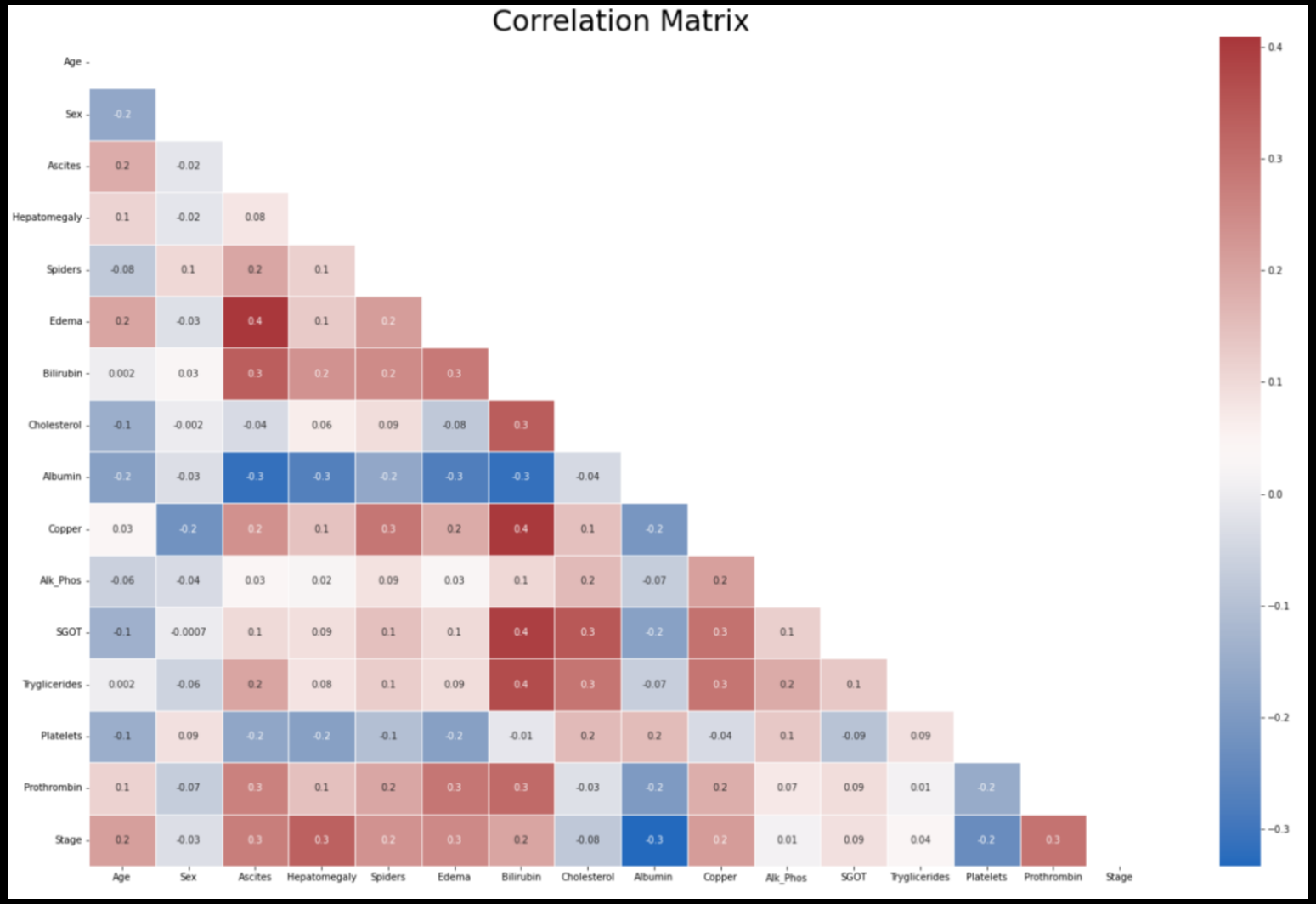


Figure 9: Distributions of the numerical features vs. the target 'Stage'.

Numerical Features

- The features follow normal distributions.
- We that patients with cirrhosis tend to be older in `Age` and have lower values for all the features.

Data Preprocessing



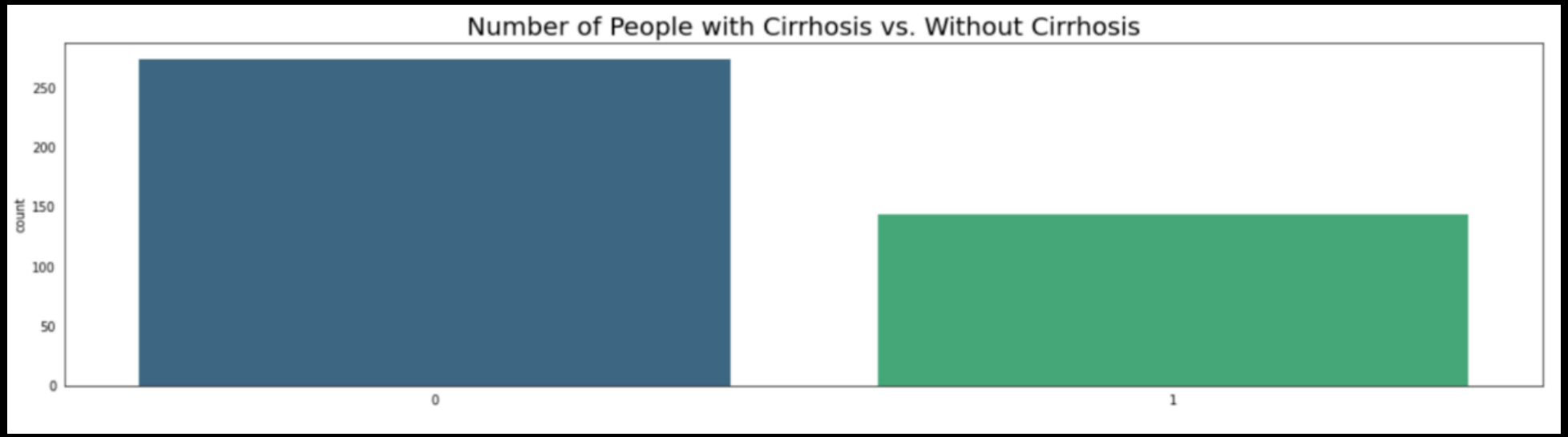
- An arbitrary correlation threshold of $r = 0.08$ was chosen as a cutoff for selecting features to include in the models (12 features included).
- The data was standardized ($\mu = 0$, $\sigma = 1$) so it would work with each model nicely.
- 80/20 Train/Test split ($n_{train} = 334$, $n_{test} = 84$).

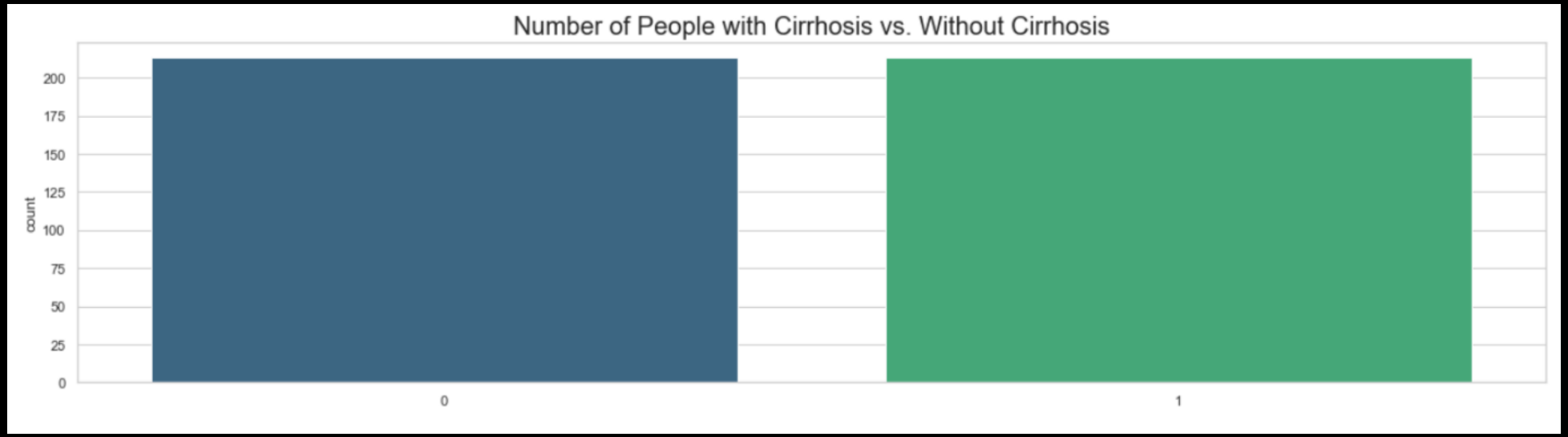
Stage	1.000000
Albumin	0.333060
Hepatomegaly	0.328234
Prothrombin	0.293754
Ascites	0.275527
Edema	0.250983
Platelets	0.236666
Spiders	0.232651
Copper	0.217266
Age	0.210092
Bilirubin	0.198645
SGOT	0.090118
Cholesterol	0.083042

Data Preprocessing

SMOTE (Synthetic Minority Oversampling TEchnique)

- SMOTE was applied on a copy of the training set.
- Creates synthetic data of the target class.
- Uses the difference between sample features multiplied by a random number between 0 and 1.
- 5 models were trained on both sets then compared.

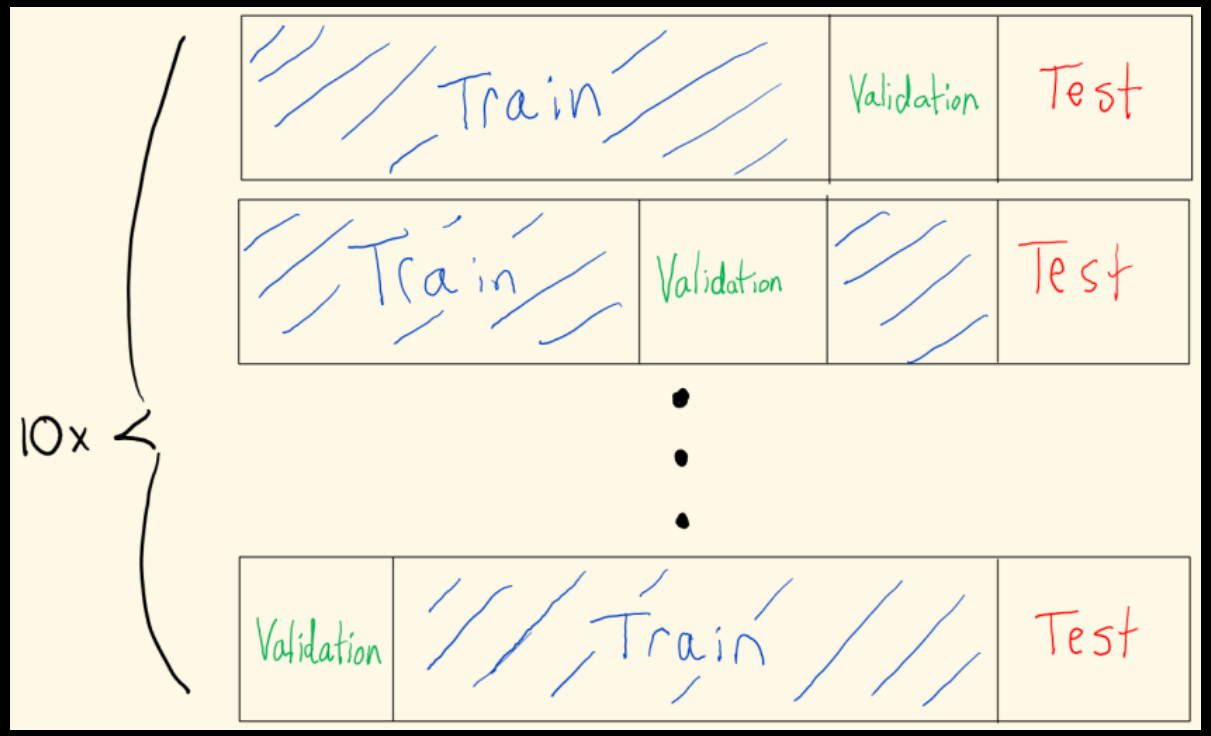


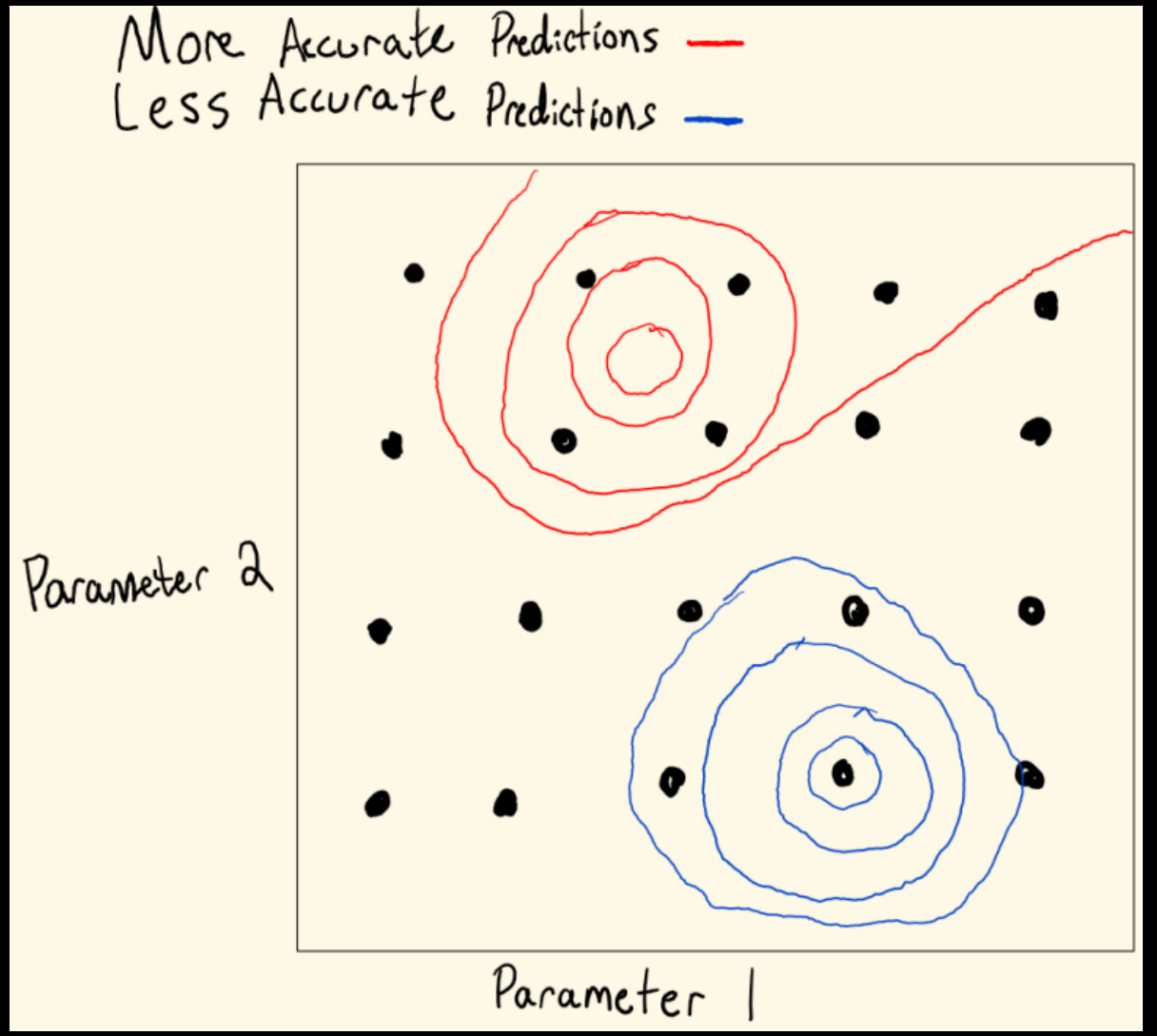


Machine Learning

Cross-Validation

- Procedure used to increase generalization potential and reduce overfitting of the models.
- An exhaustive grid-search is used to tune hyperparameters mentioned later.





Machine Learning

Pros and Cons of each Model

Logistic Regression Classifiers

- Makes no assumptions about the class distributions in feature space.
- Can interpret model coefficients as indicators of feature importance.
- Can be regularized to reduce overfitting.
- Requires that the log-likelihood odds are linearly related to the independent variables.
- Doesn't work with finding nonlinear decision boundaries.

Support Vector Machines

- Can achieve extremely high accuracy.
- Works well with many features per sample.
- Long and expensive training time.
- Prone to overfitting on datasets with lots of overlap, requiring lots of projecting into higher dimensional feature space.
- Not easily interpretable.

Machine Learning

Pros and Cons of each Model

Decision Tree Classifiers

- Implicitly interpretable and simple to understand compared to other models.
- Often suffers from overfitting.
- Generally performs worse than other models.

K-Nearest Neighbors

- Simple to understand and short training time.
- Long testing time and may overfit with large values of *k*.

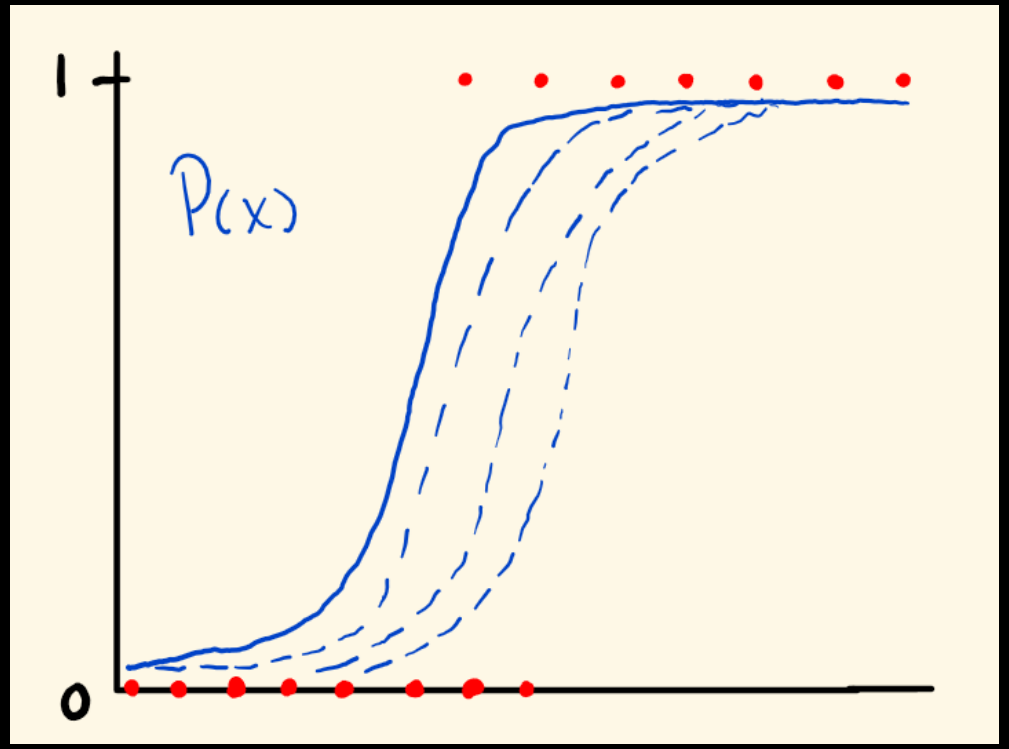
Naïve Bayes

- Fast training and testing time.
- Not sensitive to irrelevant features.
- Requires an arbitrary modification to solve the zero-probability problem.
- Assumes independence between features which isn't always true.

Machine Learning

Logistic Regression Classifier

- Models the conditional probability of a patient having cirrhosis given the data.
- Uses Maximum Likelihood Estimation to fit a logistic curve separating the target class.
- Regularization term can be added to loss function to penalize large predictor values.
- The steepness of the curve and penalty term are found via cross-validation with the training set.

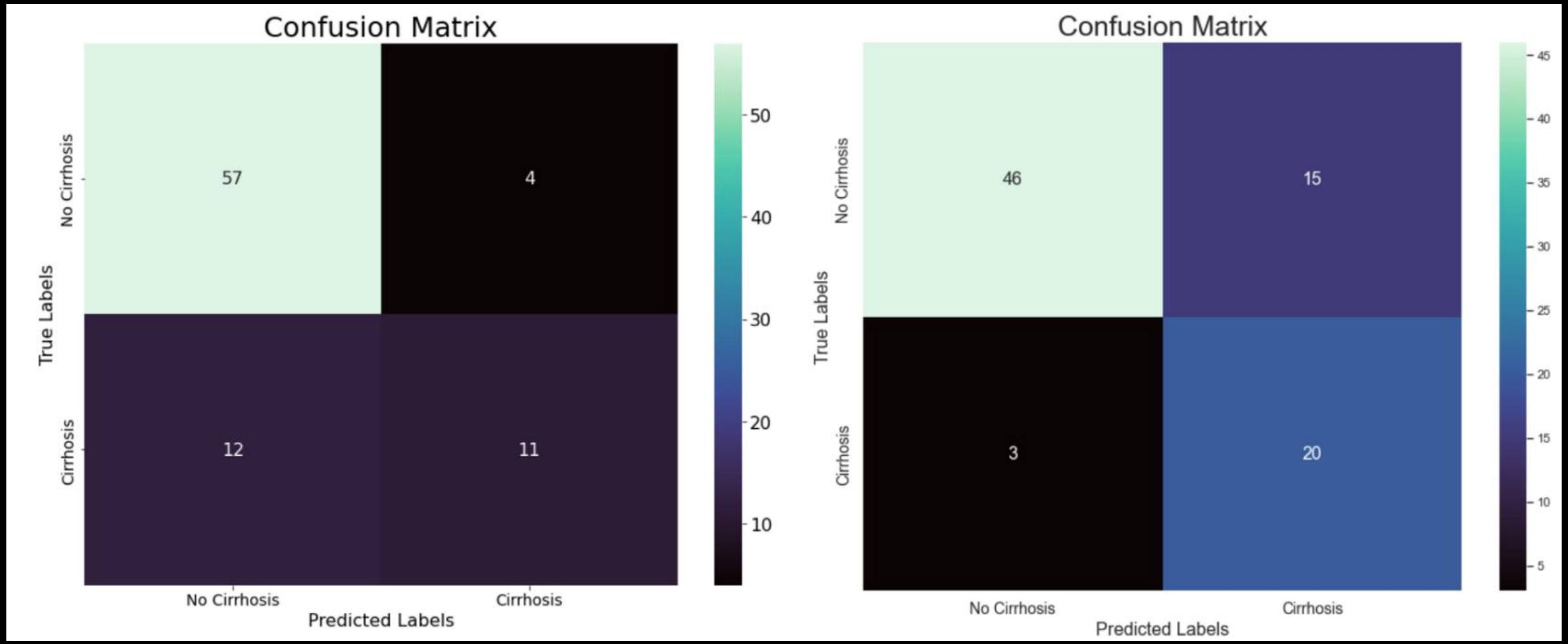


$$\text{For } i \leq n, j \leq k \text{ and } p(x_i) = \frac{1}{1+e^{-(\beta_0+\sum_{j=1}^{k}\beta_j x_{ij})}},$$

$$\min_{\beta} \; -\ln(L) = -\sum_{i=1}^{n}[y_i \ln(p(x_i)) + (1-y_i)\ln(1-p(x_i))] + \lambda \sum_{j=1}^{k} \beta_j^2 \text{ s.t. } \lambda \geq 0. \qquad (5)$$

Machine Learning

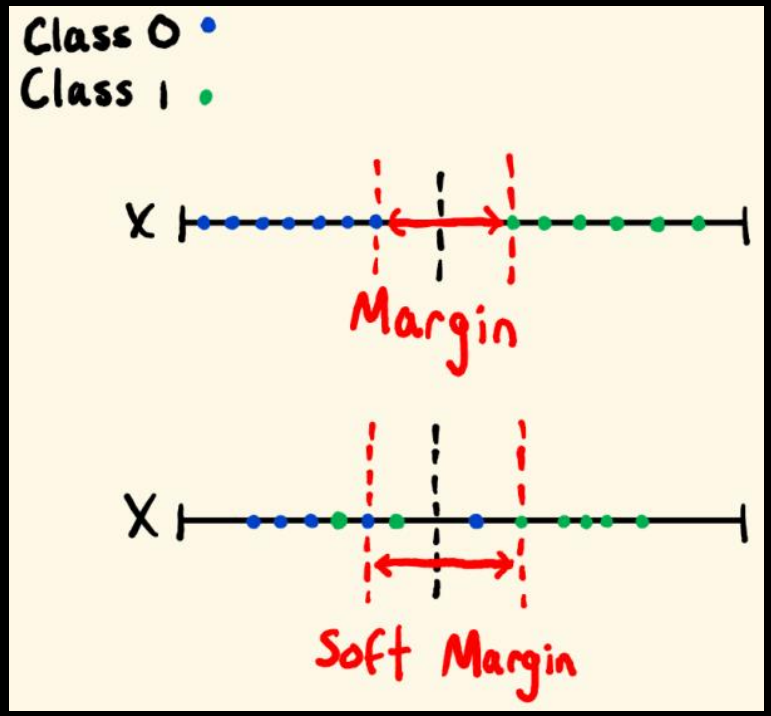
Logistic Regression Classifier



Machine Learning

Support Vector Classifier

- Attempts to find a hyperplane separating the data in feature space which maximizes the distance between the closest points of different classes.
  - Called “support vectors”
- The margins can be made softer to allow for a few misclassifications if it means increasing the overall accuracy.
- The support vectors influence the orientation of the hyperplane.
- The issue is large overlap between classes.
  - Ex/ Student X studied 3 hours and got C+
    Student Y studied 3 hours and got an A-
- SVM is used to help with overlap.



This is a quadratic convex optimization problem, so techniques like Lagrange multipliers can be used to transform it into a dual nonlinear program given by (8).

$$\max_{\beta} \; L_D = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{i'=1}^{n} \alpha_i \alpha_{i'} y_i y_{i'} x_i^T x_{i'} \tag{8}$$

$$\text{subject to} \quad \sum_{i=1}^{n} \alpha_i y_i x_i = \beta,$$

$$\sum_{i=1}^{n} \alpha_i y_i = 0,$$

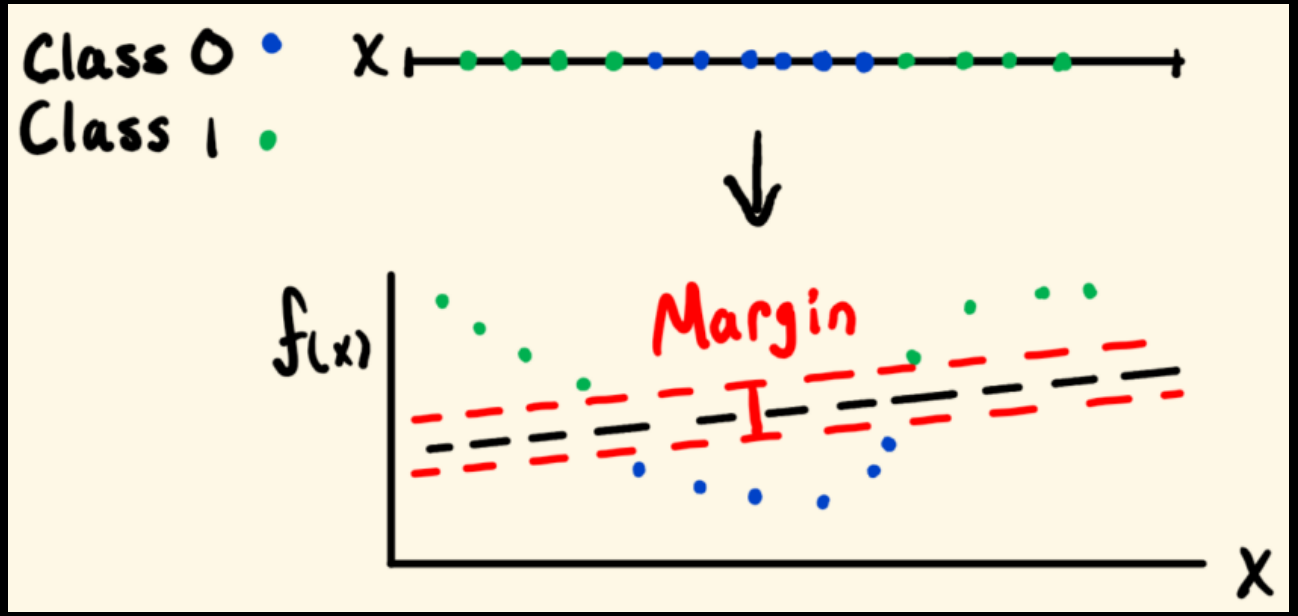
$$C - \mu_i = \alpha_i, \; \forall i \leq n,$$

$$0 \leq \alpha_i \leq C.$$

Machine Learning

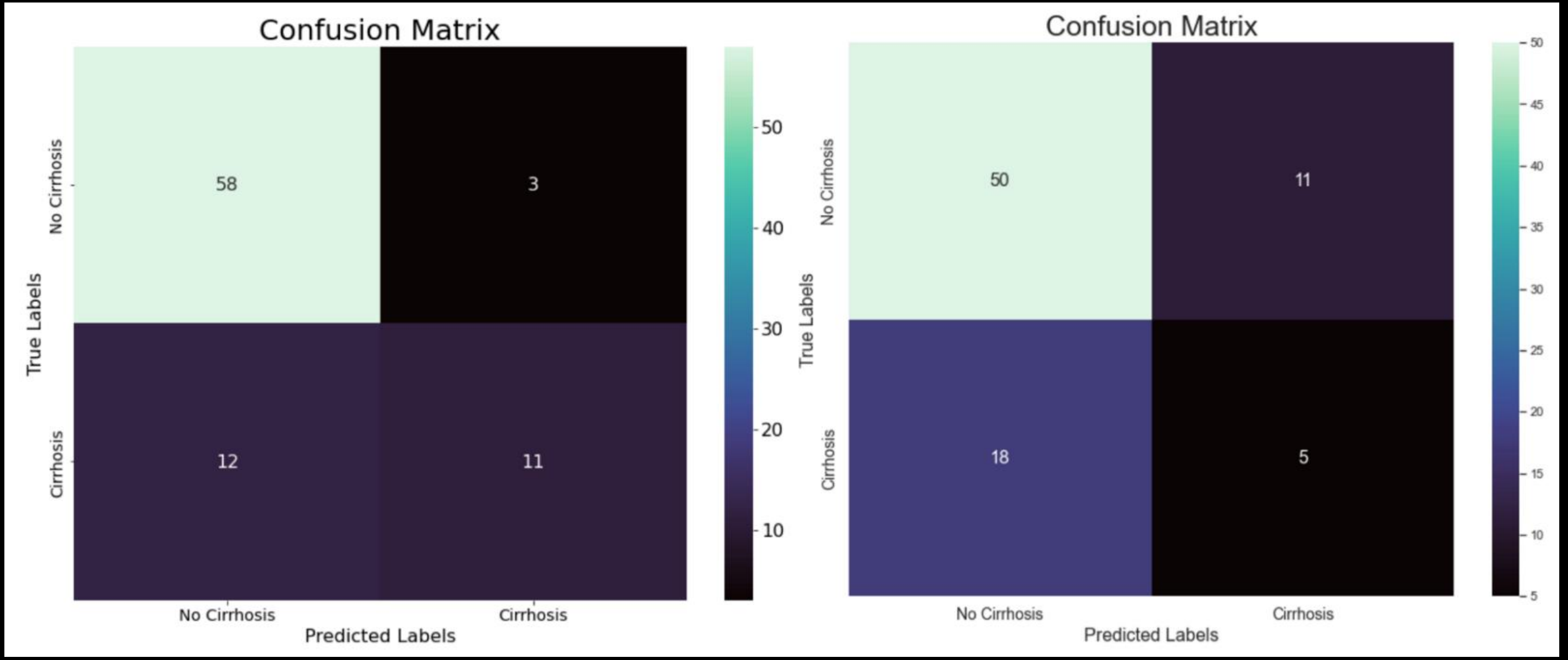
Support Vector Machine

- Projects the samples into a higher dimensional feature space which has a better chance of separating the data.
- This is done with the help of a kernel function which expresses projection function in terms of inner products, reducing the computational complexity of working in many dimensions.
- Kernels (Parameters found via cross-validation).
  - N^{th} degree polynomial
    - Smoothing parameter, C
  - Radial basis function
    - Curvature parameter, γ



Machine Learning

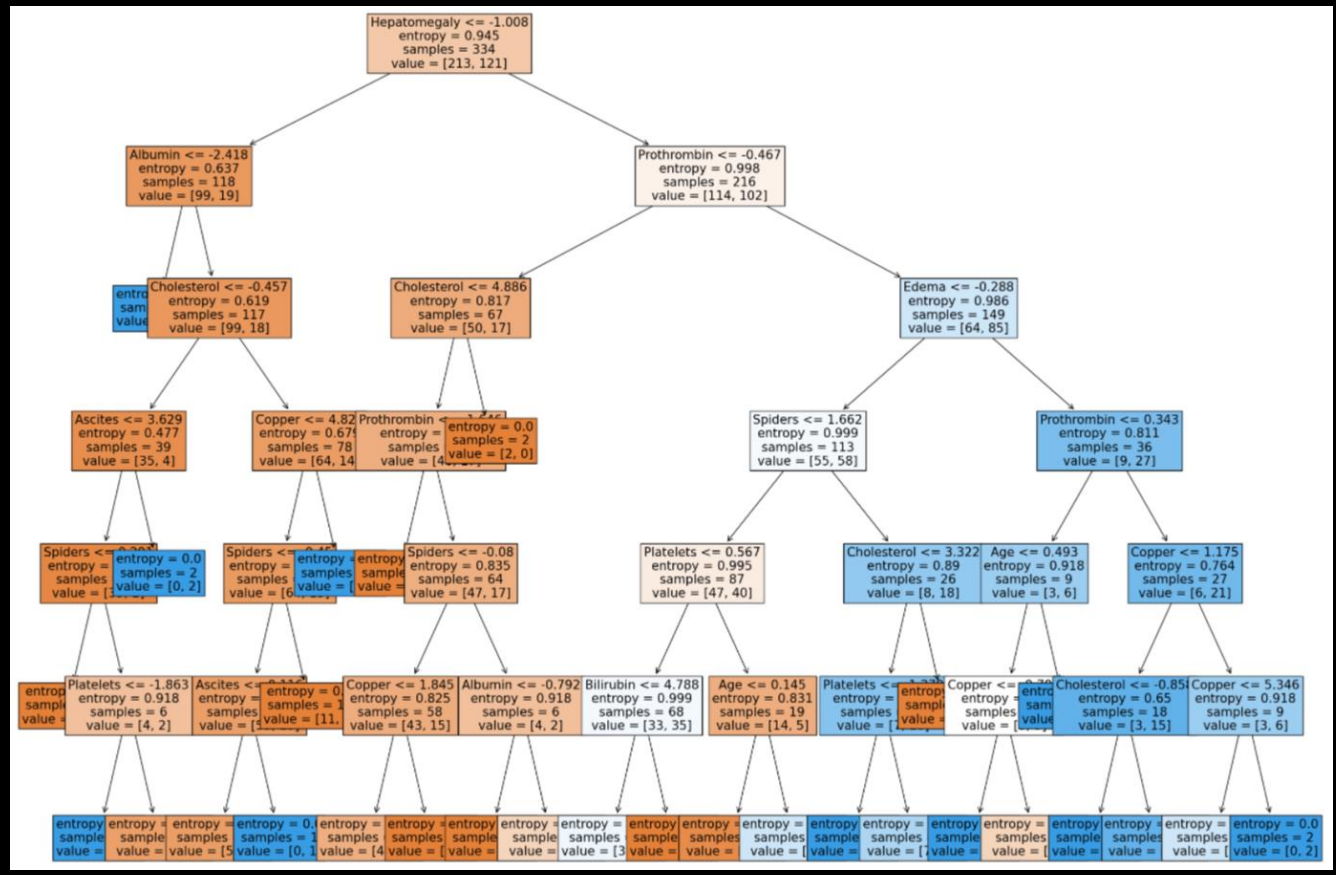
Support Vector Machine



Machine Learning

Decision Tree Classifier

- Recursively partitions the samples by feature threshold values, then deletes nodes while greedily minimizing the inhomogeneity of the final classifications.
- The inhomogeneity is measured by the Gini index.
  - Proportion of samples belonging to one class times the proportion of samples belonging to the other class.
- The goal is to minimize the cost of deleting any node in the tree.
- α determines the trade-off between the depth of the tree and prediction accuracy.
- α, the number of splits, and thresholds are found via cross-validation.



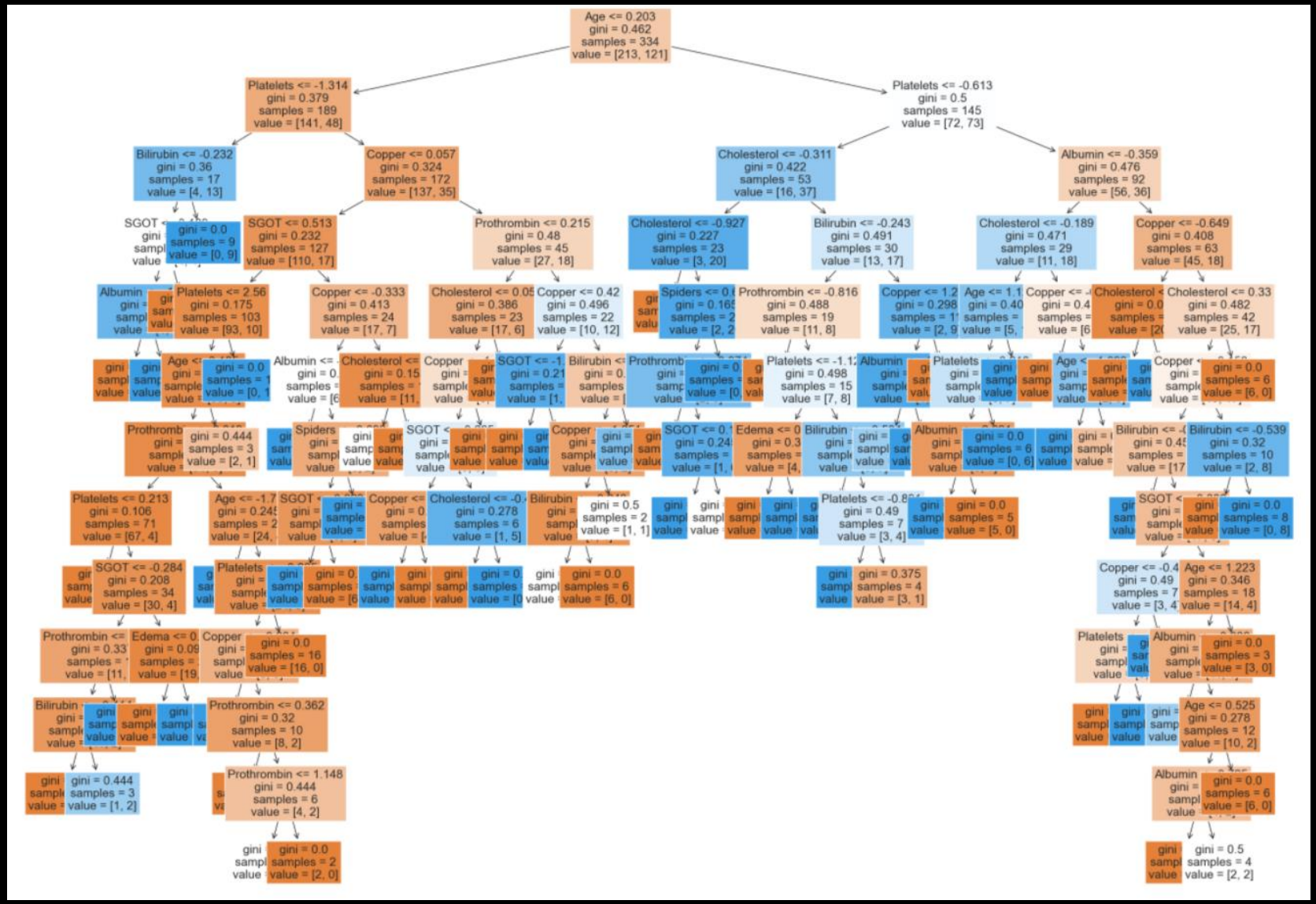
$$\min_T \; C_\alpha(T) = \sum_{m=1}^{|T|} N_m Q_m + \alpha |T| \quad \text{s.t. } \alpha \geq 0.$$

$$\text{Gini index } Q_m = \textstyle\sum_{c=0}^{1} \hat{p}_{mc}(1 - \hat{p}_{mc}).$$

Machine Learning

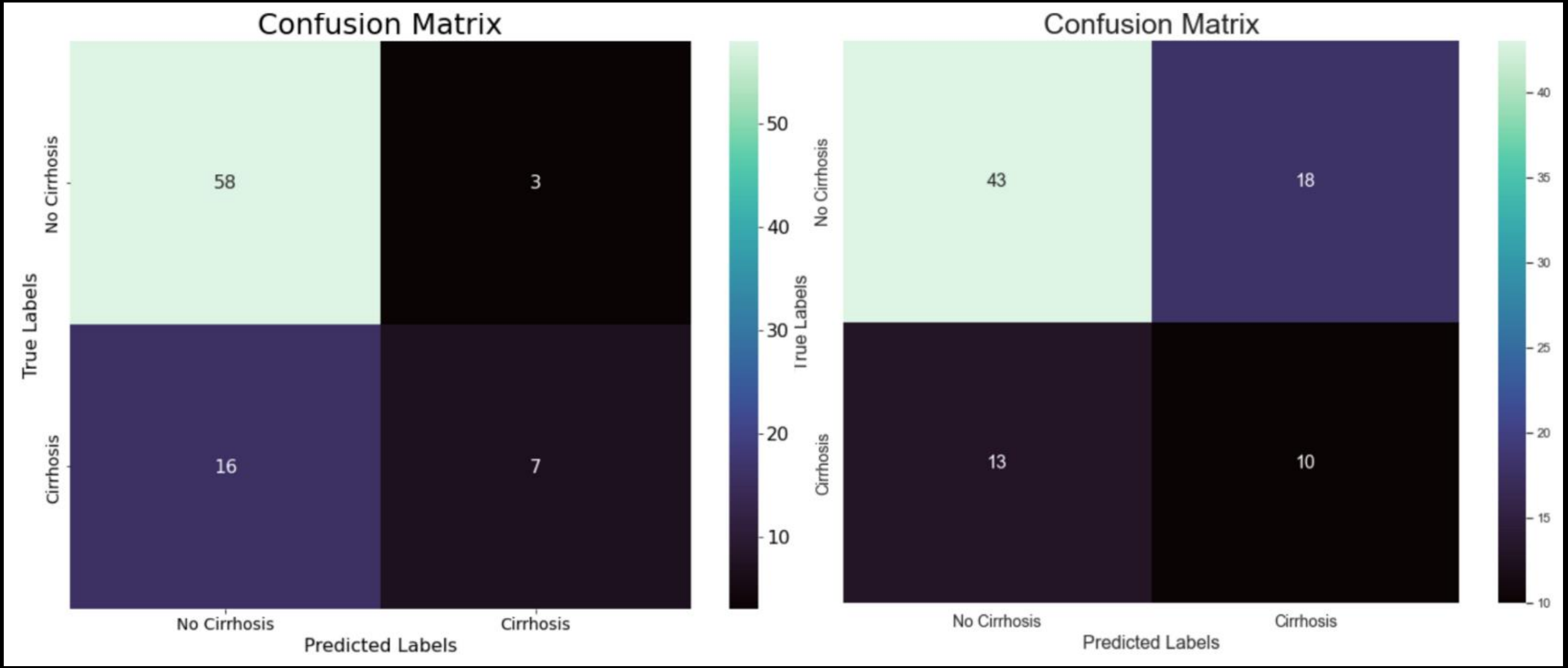
Decision Tree Classifier

- After applying SMOTE, the Decision Tree became much larger.
- Likely a result of the synthetic training data and indicative of overfitting.



Machine Learning

Decision Tree Classifier



Machine Learning

K-Nearest Neighbors Classifier

- Given a distance metric, classify new points according to the most common class among its *k*-nearest neighbors.
- Usually Euclidean or Manhattan distance is used.
- *k* and the distance metric are found via cross-validation.

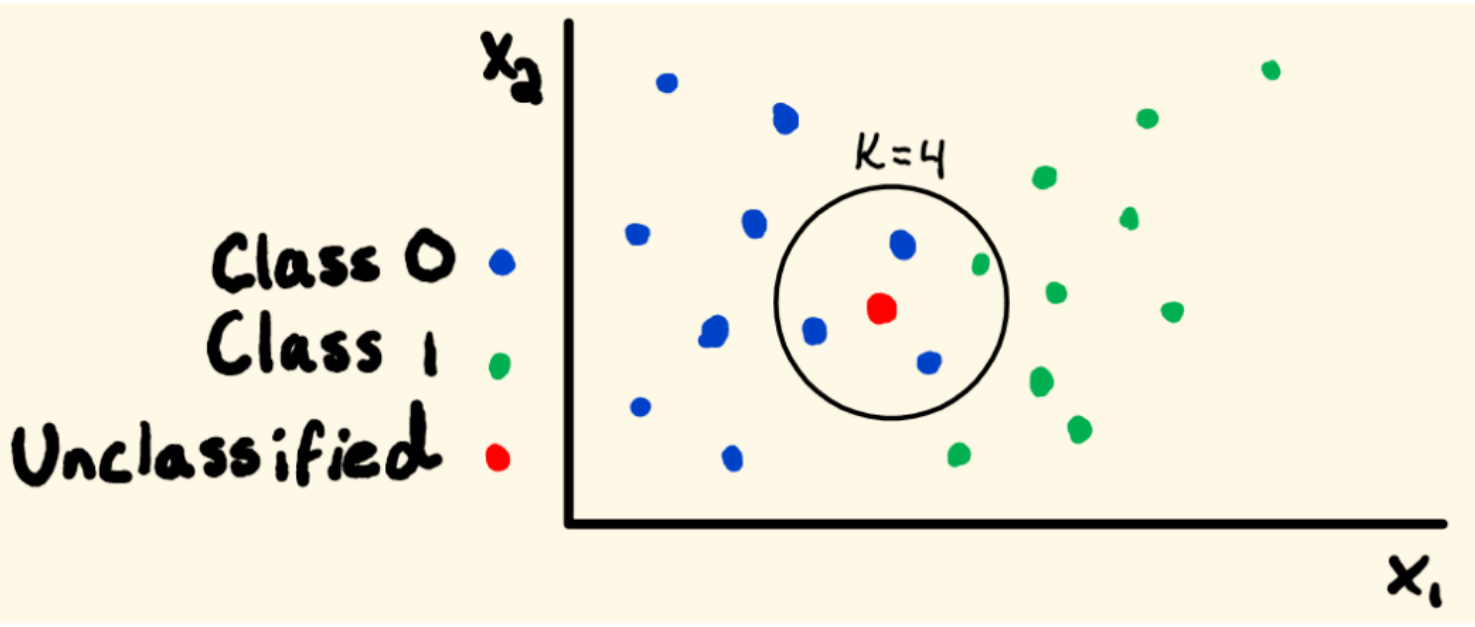
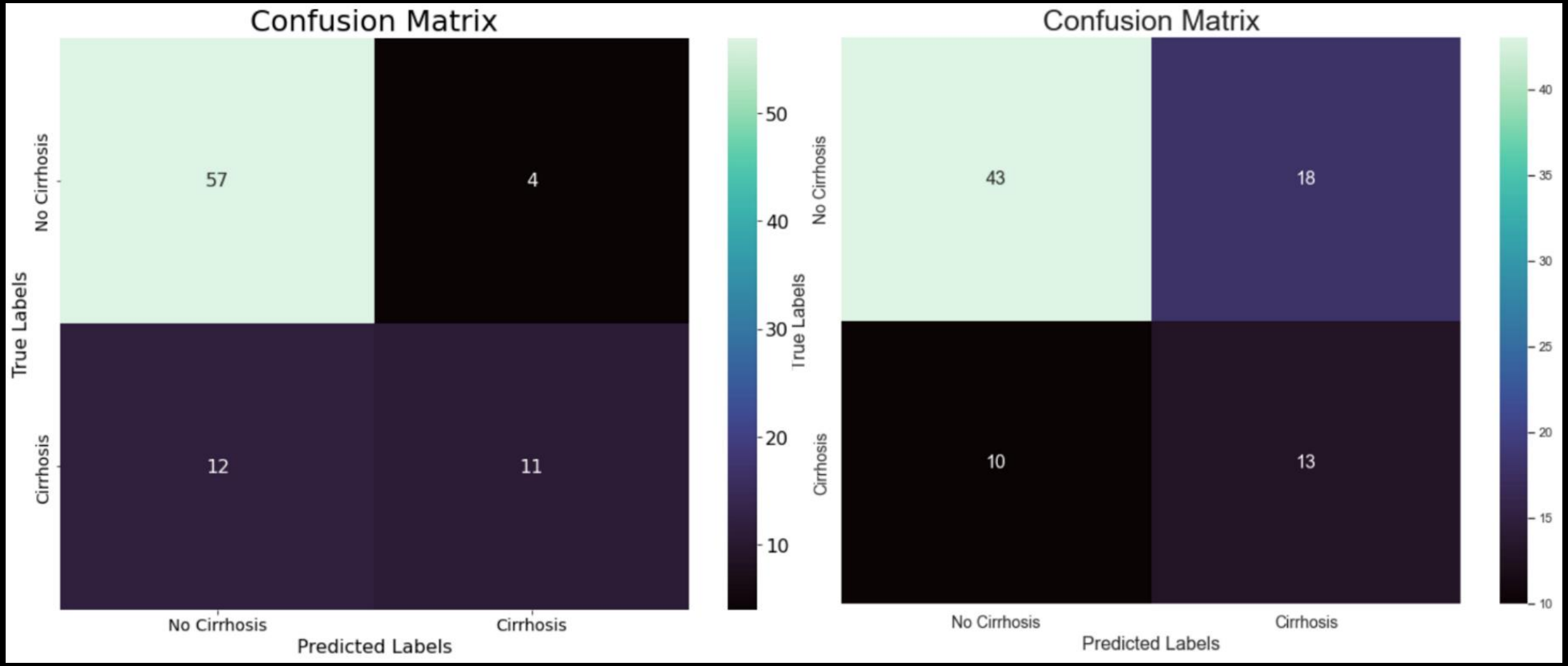


Figure 15: A KNN classifier where $k = 4$. The new point is classified as '0'.

The KNN algorithm is rather simple. Given n samples $\{(x_i, y_i) : x_i \in \mathbb{R}^m, y_i \in \{0, 1\}, i \leq n\}$, define the *Minkowski distance of order p* between points in $\mathbb{R}^m$ by $D_p(x_i, x_{i0}) = (\sum_{i=1}^{m} |x_i - x_{i0}|^p)^{\frac{1}{p}}$. Then any new sample $x_0 \in \mathbb{R}^m$ is classified as the most common among its k-nearest-neighbors, according to the Minkowski distance of order p. In other words, if $K = \{(x_i, y_i) : i \leq k \leq n\}$ is the set of samples closest to x_0 for some order p, then $y_0 = \text{argmax}_{y_i} K$. The parameters k and p are found via cross-validation and p is typically either the *Manhattan distance* ($p = 1$), or *Euclidean distance* ($p = 2$). A simple example is shown in fig. 15.

Machine Learning

K-Nearest Neighbors Classifier



Machine Learning

Naïve Bayes Classifier

- Models the joint distribution of sample *x* and target *y* and predicts the posterior probability *P(y | x)*.
- Additionally, can assume the features are conditionally independent ("Naïve").
- Complexity scales linearly in the number of features, so training is faster.

$$P(y \mid x_1, \ldots, x_n) = \frac{P(y)P(x_1, \ldots, x_n \mid y)}{P(x_1, \ldots, x_n)}$$

- If any of the probabilities $P(x_i \mid y) = 0$ for some *i*, then the entire probability becomes zero.
- This issue is solved via *smoothing* (adding a small amount to each observation, determined by cross-validation)

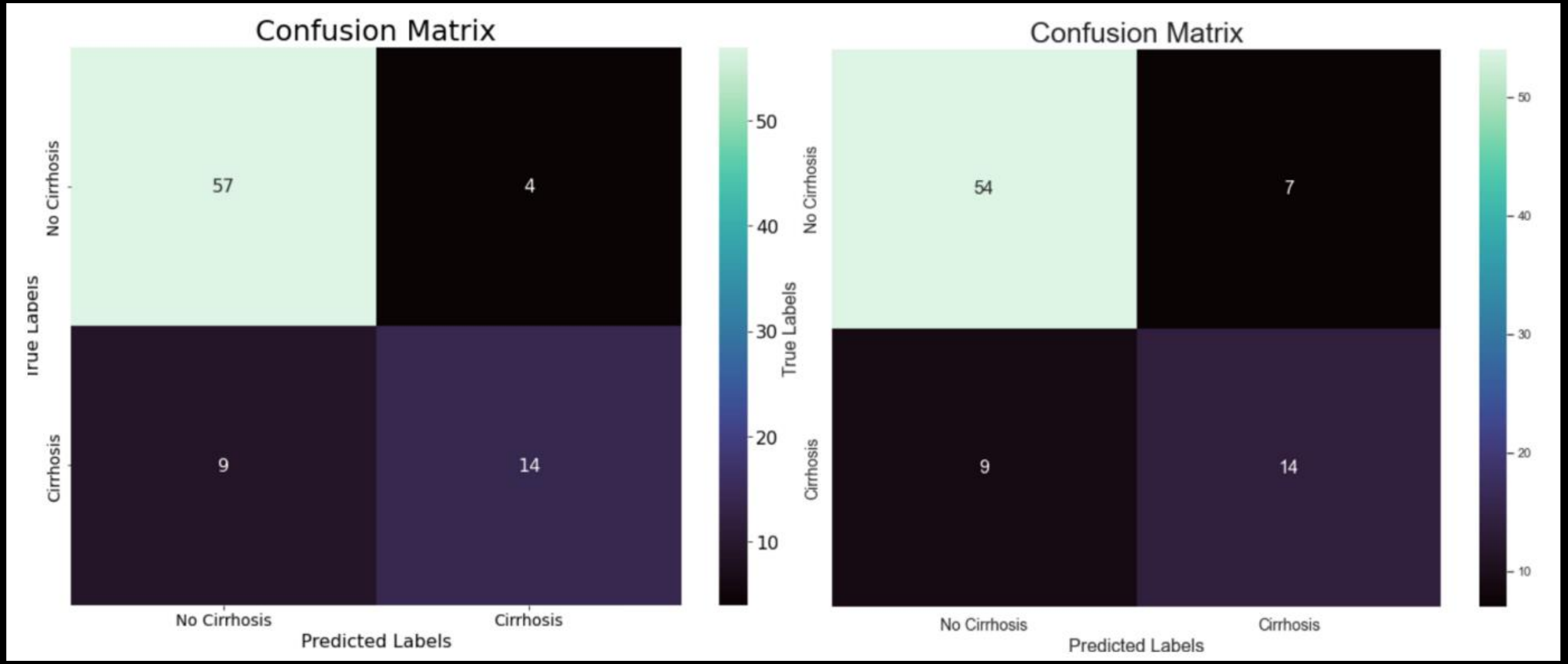
$$\text{argmax}_y \ \hat{y} = P(y) \prod_{i=1}^{n} P(x_i|y) \quad \text{for } i \leq n. \tag{10}$$

If we also assume that this conditional probability follows a normal distribution, then the problem is given by (11). $P(y)$ is estimated via cross-validation on the training set and the solution is again found using the scikit-learn library.

$$\text{argmax}_y \ \hat{y} = P(y) \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}} e^{-\frac{x_i^2}{2}} \quad \text{for } i \leq n. \tag{11}$$

Machine Learning

Naïve Bayes Classifier



Results

Model	Training Accuracy	Testing Accuracy
Logistic Regression	73.04%	80.95%
Support Vector Machine	71.56%	82.14%
Decision Tree	73.04%	77.38%
K-Nearest Neighbors	72.44%	80.95%
Naive Bayes	72.75%	84.52%

- We see an overall increase in performance on the test set than on the training set.
  - This could mean that the models have some generalization potential.
  - It could also be a manifestation of the class imbalance toward no cirrhosis.
  - The Naïve Bayes performed the best on the test set.
  - The Decision Tree performed the worst on the test set.

- The testing accuracy decreased for each model to varying degrees.
- SVM, KNN, and Decision Tree decreased.
  - Performed worse on the test set than on the training set.
- Logistic Regression and Naïve Bayes increased.
  - Performed better on the test set than on the training set.

Model	Training Accuracy (After SMOTE)	Testing Accuracy (After SMOTE)
Logistic Regression	74.19%	78.57%
Support Vector Machine	78.98%	65.48%
Decision Tree	74.92%	63.10%
K-Nearest Neighbors	80.07%	66.66%
Naive Bayes	68.45%	80.95%

Results

Model	FP	TP	FN	TN
Log-Reg	4	11	12	57
Log-Reg (SMOTE)	15	20	3	46
SVM	3	11	12	58
SVM (SMOTE)	11	5	18	50
Decision Tree	3	7	16	58
Decision Tree (SMOTE)	18	10	13	43
KNN	4	11	12	57
KNN (SMOTE)	18	13	10	43
Naïve Bayes	4	14	9	57
Naïve Bayes (SMOTE)	7	14	9	54

Model	FPR	FNR	TPR	TNR
Log-Reg	0.0656	0.5217	0.4783	0.9344
Log-Reg (SMOTE)	0.2459	0.1304	0.8696	0.7541
SVM	0.0492	0.5217	0.4783	0.9508
SVM (SMOTE)	0.1803	0.7826	0.2174	0.8197
Decision Tree	0.0492	0.6957	0.3043	0.9508
Decision Tree (SMOTE)	0.2951	0.5652	0.4348	0.7049
KNN	0.0656	0.5217	0.4783	0.9344
KNN (SMOTE)	0.2951	0.4348	0.5652	0.7049
Naïve Bayes	0.0656	0.3913	0.6087	0.9344
Naïve Bayes (SMOTE)	0.1148	0.3913	0.6087	0.8852

$$TPR = \frac{TP}{P}, \quad TNR = \frac{TN}{N}$$
$$FPR = 1 - TNR, \quad FNR = 1 - TPR$$

- Highest FPR: Decision Tree (SMOTE), KNN (SMOTE)
- Lowest FPR: SVM, Decision Tree
- Highest TPR: Logistic Regression (SMOTE)
- Lowest TPR: SVM (SMOTE)
- Highest FNR: Logistic Regression (SMOTE)
- Lowest FNR: SVM (SMOTE)
- Highest TNR: SVM, Decision Tree
- Lowest TNR: Decision Tree (SMOTE), KNN (SMOTE)

Results

Model	(Closer to 1 is better) Positive Predictive Value	(Closer to 1 is better) Negative Predictive Value	(Closer to 0 is better) False Discovery Rate	(Closer to 0 is better) False Omission Rate
Log-Reg	0.7333	0.8261	0.2667	0.1739
Log-Reg (SMOTE)	0.5714	0.9388	0.4286	0.0612
SVM	0.7857	0.8286	0.2143	0.1714
SVM (SMOTE)	0.3125	0.7353	0.6875	0.2647
Decision Tree	0.7000	0.7838	0.3000	0.2162
Decision Tree (SMOTE)	0.3571	0.7679	0.6429	0.2321
KNN	0.7333	0.8261	0.2667	0.1739
KNN (SMOTE)	0.4194	0.8113	0.5806	0.1887
Naïve Bayes	0.7778	0.8636	0.2222	0.1364
Naïve Bayes (SMOTE)	0.6667	0.8571	0.3333	0.1429

- Positive Predictive Value (PPV)
  - How much you should trust a positive prediction.
- Negative Predictive Value (NPV)
  - How much you should trust a positive prediction.
- False Discovery Rate (FDR)
  - How often false positives are made.
- False Omission Rate (FOR)
  - How often false negatives are made.

- Highest PPV: SVM
- Lowest PPV: SVM (SMOTE)
- Highest NPV: Logistic Regression (SMOTE)
- Lowest NPV: SVM (SMOTE)
- Highest FDR: SVM (SMOTE)
- Lowest FDR: SVM
- Highest FOR: SVM (SMOTE)
- Lowest FOR: Logistic Regression (SMOTE)

$$PPV = \frac{TP}{TP + FP}, \quad NPV = \frac{TN}{TN + FN}$$

$$FDR = 1 - TNR, \quad FOR = 1 - NPV$$

PPV

Results

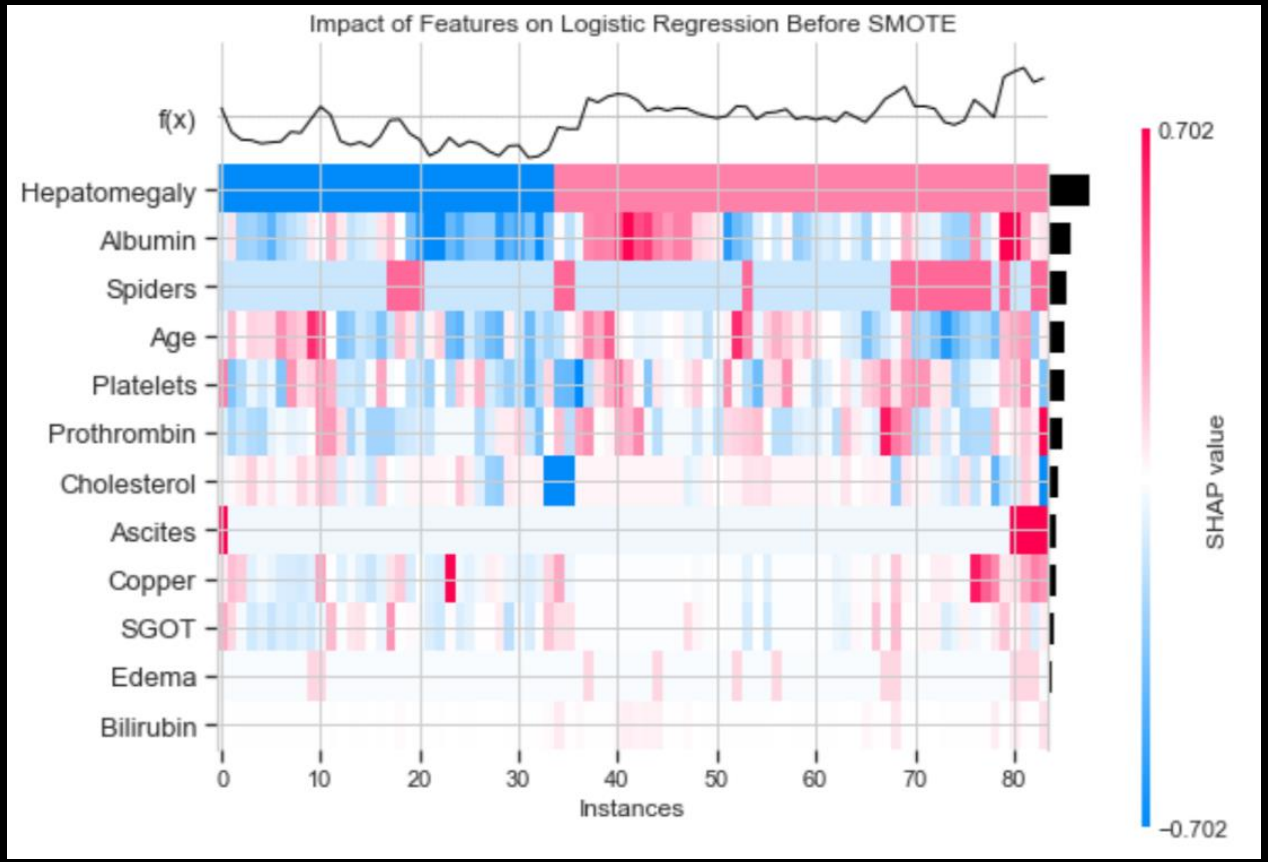
Model	(Larger is better) Positive Likelikhood Ratio	(Smaller is better) Negative Likelihood Ratio	(Closer to 1 is better) F-1 Score
Log-Reg	7.2935	0.5584	0.5789
Log-Reg (SMOTE)	3.5362	0.1730	0.6897
SVM	9.7246	0.5487	0.5946
SVM (SMOTE)	1.2055	0.9548	0.2564
Decision Tree	6.1884	0.7316	0.4242
Decision Tree (SMOTE)	1.4734	0.8018	0.3922
KNN	7.2935	0.5584	0.5789
KNN (SMOTE)	1.9155	0.6168	0.4815
Naïve Bayes	9.2826	0.4188	0.6829
Naïve Bayes (SMOTE)	5.3043	0.4420	0.6364

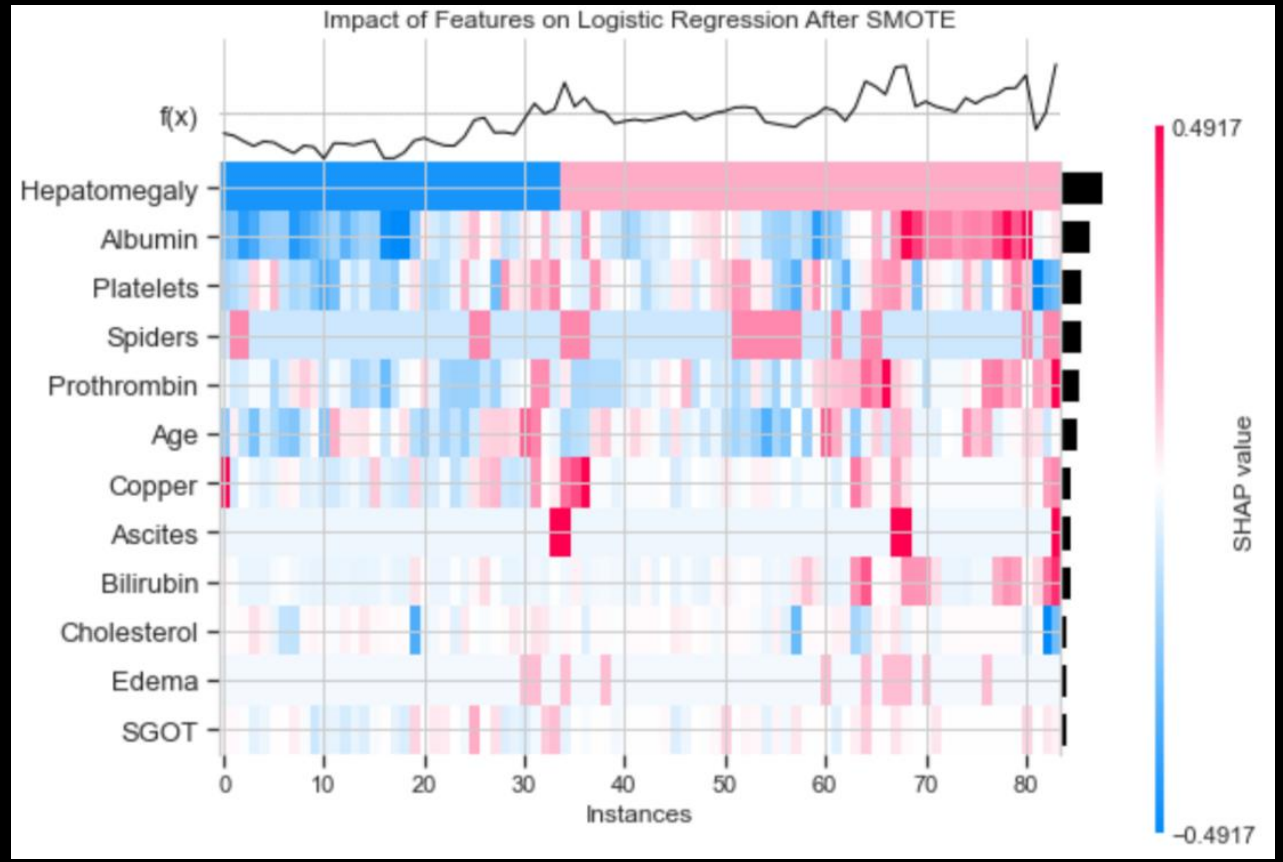
- Positive Likelihood Ratio
  - The probability that someone who has cirrhosis tests positive / The probability that someone who does not have cirrhosis tests positive.
- Negative Likelihood Ratio
  - The probability that someone who has cirrhosis tests negative / The probability that someone who does not have cirrhosis tests negative.
- F1 Score
  - Harmonic mean between PPV and TPR

- SVM performed the best at classifying cirrhosis while minimizing false positives.
- Logistic Regression performed the best at classifying cirrhosis while minimizing false negatives.
- Naïve Bayes performed the best overall.

$$PLR = \frac{TPR}{FPR}, \quad NLR = \frac{FNR}{TNR}, \quad F1\,Score = 2 \cdot \frac{PPV \cdot TPR}{PPV + TPR}$$

Results

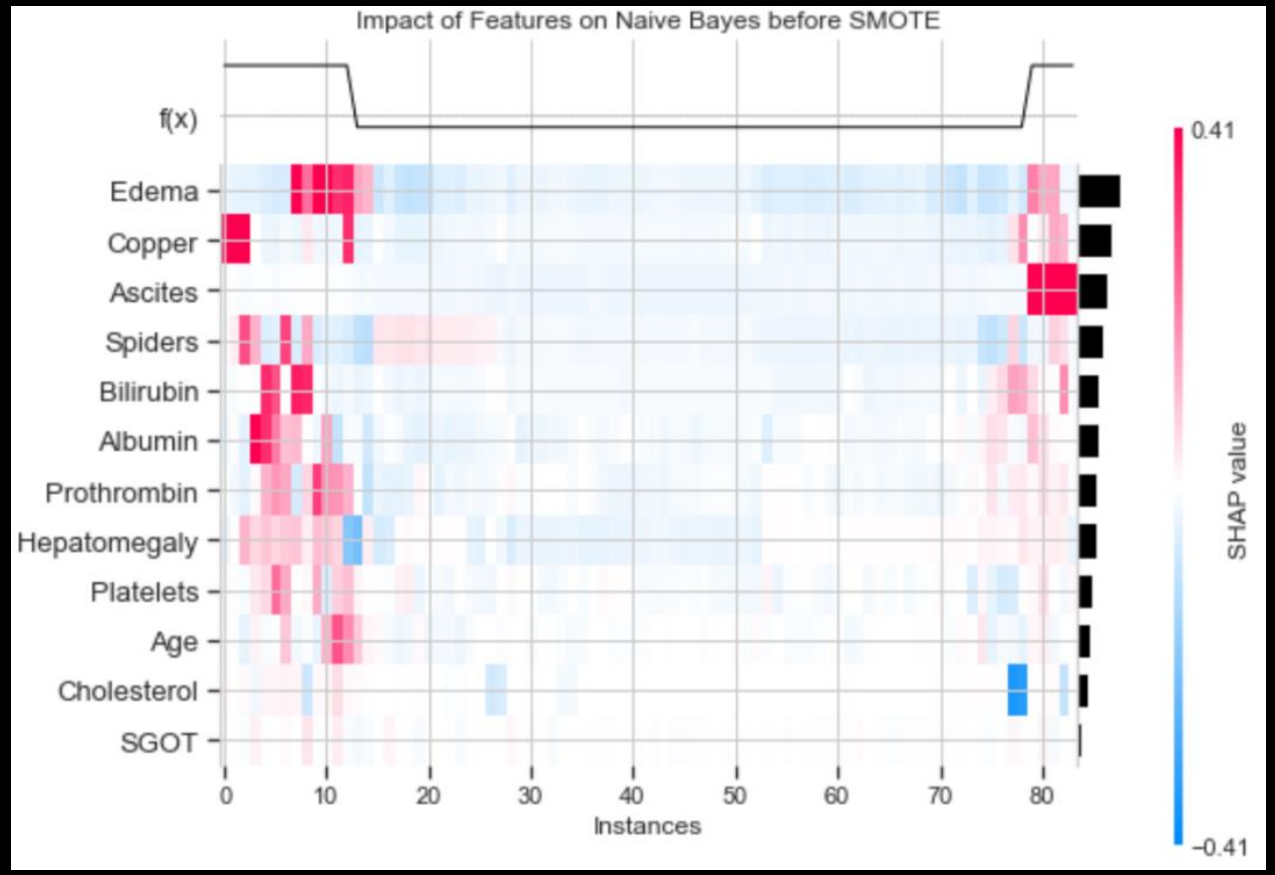


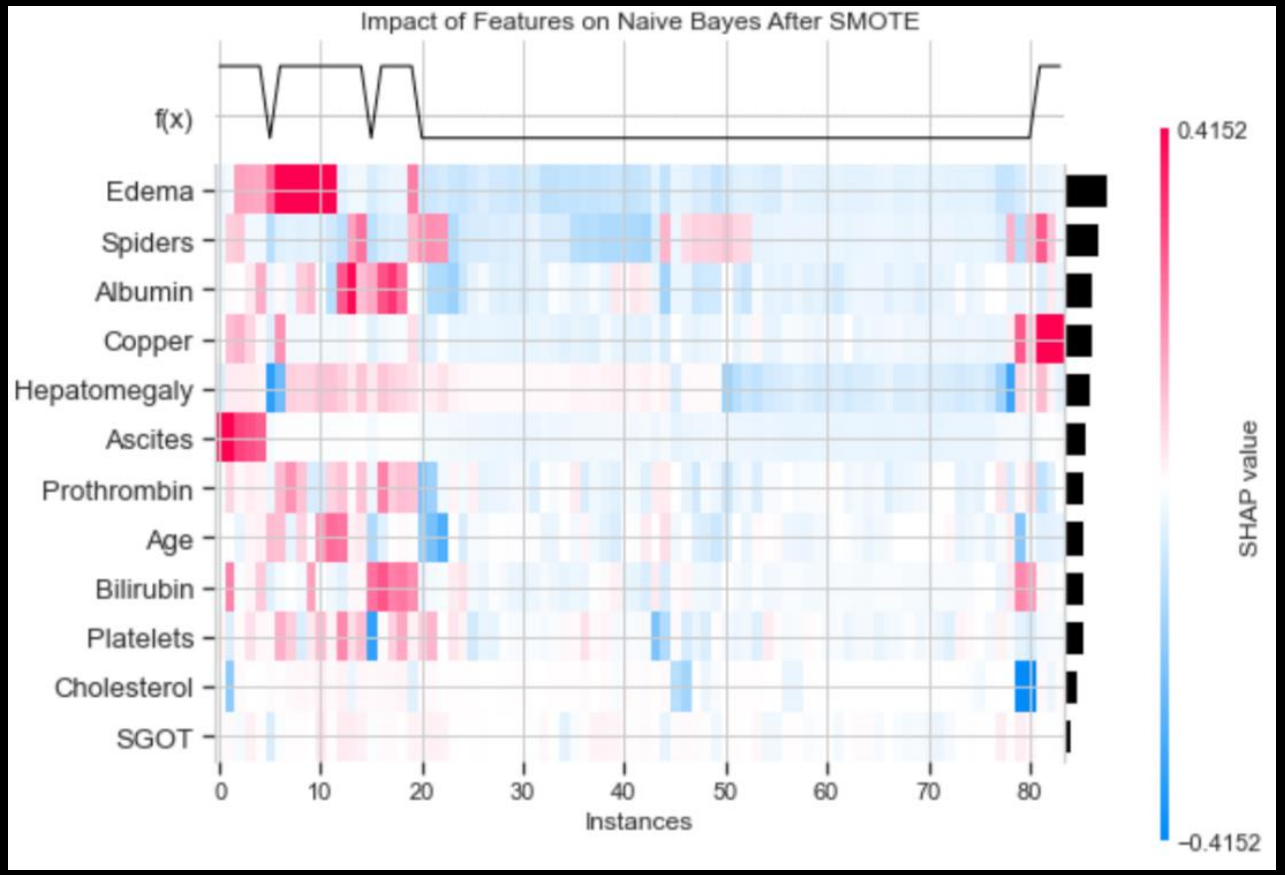


How did SMOTE change the feature importance of the Logistic Regression model?

- SMOTE decreased the importance of `Spiders`, `Age`, `Cholesterol`, and `SGOT`.
- SMOTE increased the importance of `Platelets`, `Prothrombin`, `Copper`, and `Bilirubin`.
- `SGOT` and `Edema` barely contribute before and after SMOTE.

Results





How did SMOTE change the feature importance of the Naïve Bayes model?

- SMOTE decreased the importance `Copper`, `Ascites`, `Bilirubin`, and `Platelets`.
- SMOTE increased the importance of `Spiders`, `Albumin`, `Hepatomegaly`, and `Age`.
- `SGOT` barely contributes before and after SMOTE.

Discussion

- Sources of error
  - Imputation of missing data reducing the variance of the features, introducing bias.
  - Class imbalance in target class leading to high false negative rates.
  - KNN and SVM likely overfit the synthetic data.
- Which model is “best”?
  - The Decision Tree is most interpretable in general and was tied with SVM for lowest false positive rate (4.92%).
  - Logistic Regression (SMOTE) had the highest true positive rate (86.96%).
  - Naïve Bayes had the best overall accuracy before and after SMOTE (84.52%, 80.95%).
- What would I have done differently?
  - Spend more time on rectifying missing data.
  - Use a more robust feature selection process.
  - Reduce the dimensionality of the data for training the SVM.
  - SMUTE (Synthetic Majority Undersampling TEchnique).
  - Test on another dataset.
  - Try another model (Random Forest Classifier)

Model	Test Accuracy (Before SMOTE)	Test Accuracy (After SMOTE)
Logistic Regression	80.95%	78.57%
Support Vector Machine	82.14%	65.48%
Decision Tree	77.38%	63.10%
K-Nearest Neighbors	80.95%	66.66%
Naive Bayes	84.52%	80.95%

References

[1] Javad Hassannataj Joloudari, Hamid Saadatfar, Abdollah Dehzangi, and Shahaboddin Shamshirband. Computer-aided decision-making for predicting liver disease using pso-based optimized svm with feature selection. *Informatics in medicine unlocked*, 17:100255, 2019.

[2] Moloud Abdar, Neil Yuwen Yen, and Jason Chi-Shun Hung. Improving the diagnosis of liver disease using multilayer perceptron neural network and boosted decision trees. *Journal of Medical and Biological Engineering*, 38(6):953–965, 2018.

[3] E Rolland Dickson, Patricia M Grambsch, Thomas R Fleming, Lloyd D Fisher, and Alice Langworthy. Prognosis in primary biliary cirrhosis: model for decision making. *Hepatology*, 10(1):1–7, 1989.

[4] Bernd H Markus, E Rolland Dickson, Patricia M Grambsch, Thomas R Fleming, Vincenzo Mazzaferro, Goran Bo G Klintmalm, Russell H Wiesner, David H Van Thiel, and Thomas E Starzl. Efficacy of liver transplantation in patients with primary biliary cirrhosis. *New England Journal of Medicine*, 320(26):1709–1713, 1989.

[5] Georg Hoffmann, Andreas Bietenbeck, Ralf Lichtinghagen, and Frank Klawonn. Using machine learning techniques to generate laboratory diagnostic pathways—a case study. *J Lab Precis Med*, 3:58, 2018.

[6] Reza Safdari, Amir Deghatipour, Marsa Gholamzadeh, and Keivan Maghooli. Applying data mining techniques to classify patients with suspected hepatitis c virus infection. *Intelligent Medicine*, 2022.

[7] fedesoriano. Cirrhosis prediction dataset. https://www.kaggle.com/datasets/fedesoriano/cirrhosis prediction-dataset. Accessed March, 2022.

[8] Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009.

[9] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[10] Christoph Molnar. Shapley values. https://christophm.github.io/shapley. Accessed April, 2022.

Jupyter Notebook URL: *https://github.com/barstow2/mth496*