Capstone Project

Course Code: PRJ 600

Course Title: Capstone Project

Submitted to: Musabbir Hasan Sammak

Submitted by: Siyam Sajnan Chowdhury

ID: 231000361

Comparative Analysis of Machine Learning Models, Artificial Neural Network and Ensembles on Bladder Cancer Survival Prediction

1. Introduction

Bladder cancer is one of the deadliest forms of cancer in the world. It is a form of tumor that starts to grow in the urinary bladder lining, which leads to uncontrollable and abnormal growth of cells withing the bladder. It is the 6th deadliest form of cancer in the US with around 82,290 new cases in 2023 and 16,710 deaths [1]. This form of cancer has a 5-year relative survival rate of around 79% [2]. The etiology of bladder cancer depends upon a large variety of factors including genetic factors, environmental factors, and even personal and lifestyle factors.

Timely detection and diagnosis are crucial to save the lives of people as the chances of survival increases manifold when timely interventions are carried out. The prediction of survival could be an essential tool that diagnostic medical practitioners could use in order to customize and tailor the best treatment strategy for individual patients assessing their individual cases and keeping the risks in mind. They can use this predictive model as a tool that they can utilize to enhance the effectiveness and timely administration of treatment plans and help keep any potential side effects in check.

It also plays a crucial role in making the patients aware of their situation and better help cope with the effects of it, and they could use it to decide what recourse to plan. It helps them with informed decision-making, and it helps their families to comprehend the reality of the prognosis and helps make them realistic plans and decisions.

2. Problem Statement & Objective

Predicting the possibility of survival when affected by bladder cancer, given certain features, would have a massive impact on both the diagnostic and prognostic aspects of the medical work as well as help manage expectation for the person affected by it. In this work, I have conducted a comparative study of different machine learning models. I have implemented the machine learning models Random Forest (RF) [3], Support Vector Machines (SVM) [4] with different kernels,

AdaBoost [5], K-Nearest Neighbor (KNN) [6], Extreme Gradient Boosting (XGBoost) [7], and the Artificial Neural Network (ANN) [8] and then created an ensemble model incorporating the voting classifier [9]. The models were trained and tested on the Bladder Cancer (MSK Cell Report) dataset [10] containing information about different features, both genetic as well as physiological.

3. Research Question

In this section, I will state and articulate the research enquiries that underpin this investigation. My investigation into various machine learning and deep learning models as well as ensemble models is driven by a list of comprehensive questions that forms the basis of the objective of this project.

My first research question is how different machine learning models would perform in terms of predicting the survival of people suffering from bladder cancer.

My second research question is to see how the performance of the different machine learning models varied across the models SVM, RF, XGBoost, AdaBoost and KNN as well as Artificial Neural Network across the different metrics accuracy, precision, recall and f1-score in the domain of bladder cancer survival prediction.

4. Literature Review

Hasnain et al. [11] compared different machine learning models such as SVM, Adaboost and Random Forest to predict oncologic outcomes in patients undergoing radial cystectomy, which is, the possibility of recurrence of cancer and survival.

Tsai et al. [12] used a dataset with different types of cancers and trained the models to predict bladder cancer amongst those cancer. He first went ahead and did a two-step feature selection process combined with WEKA and forward selection. He used the machine learning models decision tree, random forest, SVM, XGBoost and lightGBM. They found the highest accuracy for lightGBM of 86.9%

Tokuyama et. Al [13] predicted the recurrence of non-muscle invasive bladder cancer by using nuclear features. They strived to predict early recurrence. The used the models Support Vector Machines and Random Forest and achieved 90% accuracy with SVM and an accuracy of 86.7% with Random Forest.

5. Methodology

In this section, I shall describe the dataset and explain the methodology to process the dataset and description of the models that I ran to predict the survival of bladder cancer.

a. Dataset Description

The dataset I used is obtained from the cbioportal website which is the Bladder Cancer (MSK Cell Report) dataset [10] containing 1659 samples. The features of the default dataset are as follows:

Study ID, Patient ID, ID, Age at Diagnosis, Age at Which Sequencing was Reported (Years), Cancer Type, Cancer Type Detailed, Ethnicity Category, Fraction Genome Altered, Gene Panel, Intravesical Treatment, Metastatic Site, Met Location, MSI Score, MSI Type, Mutation Count, Oncotree Code, Overall Survival (Months), Overall Survival Status, Pediatric Case Indicator, Primary Tumor Site, Race Category, Religion, Sample Class, Number of Samples Per Patient, Sample coverage, Sample Type, Sex, Smoker, Somatic Status, Specimen Stage, Systemic Treatment, TMB (nonsynonymous), Tumor Purity, Treatment between Pri-Met sample collection.

The target variable is the Overall Survival Status which includes values 0 and 1, with 0 meaning living and 1 meaning deceased.

I used two different sets of datasets – one was the default dataset and another one with the majority target classes downsampled.

b. Data Preprocessing

I preprocessed the data first by dropping the columns that either had no effect on the predictions or had way too many missing values. I then removed the missing values or ones with the values NAN.

I then removed all the unknown values from each and every column and used label_encoder to encode the categorical data.

I then applied random downsampling and created a new separate dataset to separately run my models again.

c. Models

Machine learning models are computational systems that has the potential to identify trends and potential structures and make decisions without the need to be explicitly programmed and not

requiring human intervention. They use the data provided and leverage those to learn about the trends and patterns of the given data thus improving upon it over time enhancing the performance.

The models used in this project will be briefly discussed as follows:

i) Random Forest

Random forest is a very capable machine learning algorithm which can solve classification and regression tasks very accurately and it improves predictive robustness. It is an ensemble learning model where the term “forest” in random forest refers to a collection of “trees”, which stands for decision trees.

Random forest algorithm starts by creating many different bootstrapped datasets. Bootstrapped datasets are constructed by randomly picking samples from the data with replacement (after a sample is chosen, it is put back in the original dataset so one sample can be chosen more than once). This creates diversity and variance in the trees and reduces the chance of overfitting and lack of generalization in decision trees.

After this, a preset number of features are selected at random at each step to create decision trees. This introduces randomization of features thus decreasing the chances of the trees becoming highly correlated.

The results from the decision trees are then combined, or aggregated. For classification problem, this is generally done using majority voting and for regression, finding the average of all the predictions are used. The architecture is illustrated by figure 1.

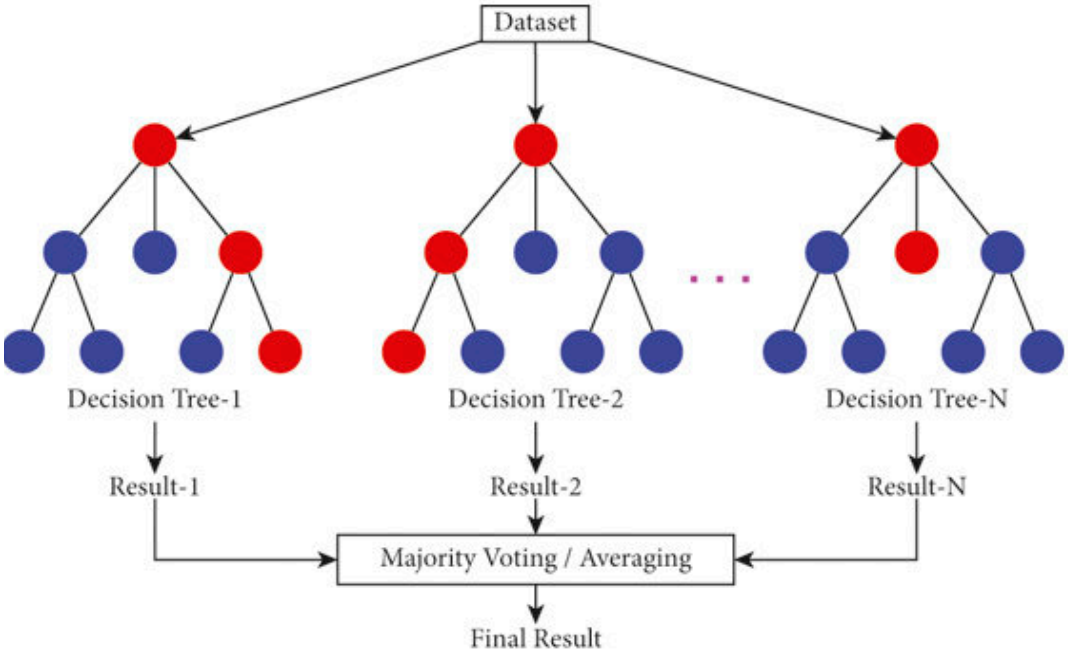

Figure 1: Random Forest architecture [14]

ii) SVM

Support vector machines, most commonly known as SVMs, are one of the most powerful machine learning models available for both classification and regression tasks. It is one of those machine learning models that consistently perform on par with deep learning models and in some cases, even outperforms them. It is a supervised learning model where the different data point represented in a specific dimension is used to find the optimal hyperplane or decision boundary that categorizes the different data points even if it is required to be represented in a higher-dimensional space. It does this by finding the points that are near the supposed decision boundary or the hyperplane, known as the support vectors.

For data is generally represented as coordinates in a high-dimension space where each dimension represents a feature in the dataset. The goal is to find the best decision boundary or hyperplane that maximizes the margin between the hyperplane and the data point of each class that is closest to the hyperplane, or the support vectors. The wider the margin, the better the SVM model's capacity to generalize to new and unseen data.

The functions of SVM are illustrated in Figure 2:

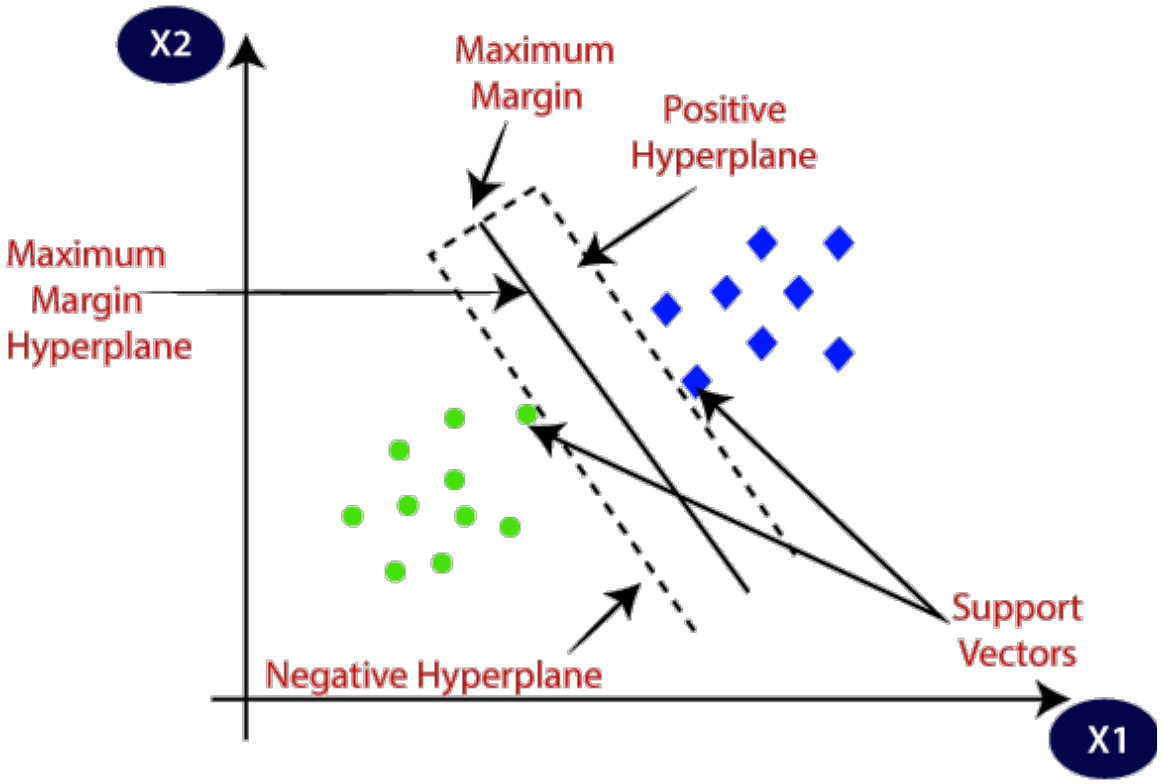

Figure 2: SVM hyperplane and margin [15]

iii) AdaBoost

AdaBoost, or adaptive boosting, is a tree-based ensemble machine learning model. The main goal is to enhance the performance of the weak learners.

AdaBoost is also an ensemble learning algorithm that works as a strong classifier for binary prediction task. It does so by combining the predictions of multiple weak learners. As opposed to random forest where the forest referred to a collection of decision trees, AdaBoost can be said to be a collection of stumps, or a decision tree based on just one feature. It can be tweaked for multi-class classification and regression as well.

The architecture of AdaBoost is illustrated in figure 3:

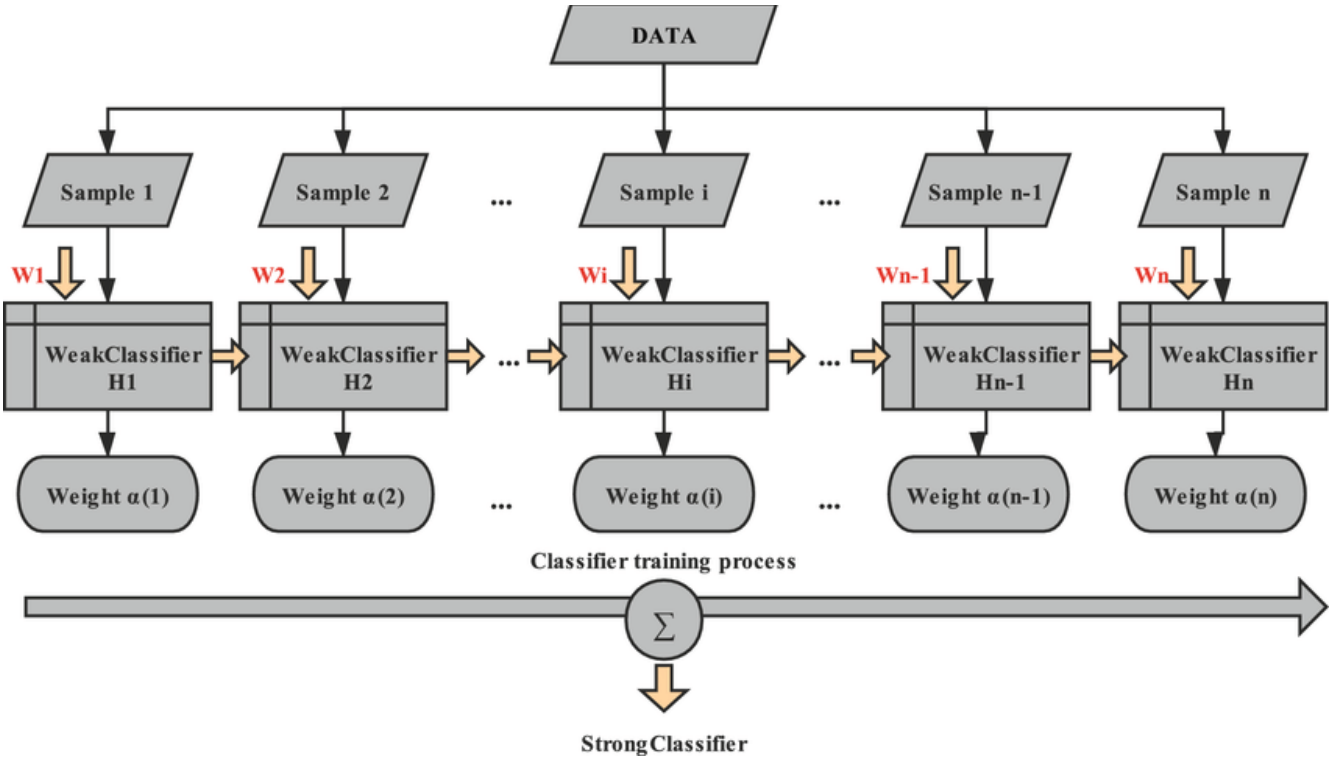

Figure 3: AdaBoost architecture [16]

iv) XGBoost

XGBoost, which stands for Extreme Gradient Boosting algorithm, is a decision tree-based model that is built on the gradient boosting model, and it is designed to minimize the time to converge with high efficiency and accuracy. It is a type of boosting algorithm where the decision trees with higher loss, otherwise known as the weak learners, are sequentially appended to the existing ensemble in an attempt to right the errors of the current ensemble. It is a powerful algorithm that is used for both classification and regression tasks and have been used in numerous different domains.

The gradient boosting algorithm starts out by initializing a prediction, which could be the mean of a target in regression tasks and log-odds of the distribution for classification tasks. This is used to compute the initial residuals. Based on this the first tree is formed, which is also the first weak learner of the model.

Then regularization is carried out followed by pruning to reduce overfitting and the model memorizing the outcomes. The prediction is then updated and the whole process is repeated until the minimum of a gradient is reached or until a predefined number of boosting is reached.

The architecture of XGBoost is presented below:

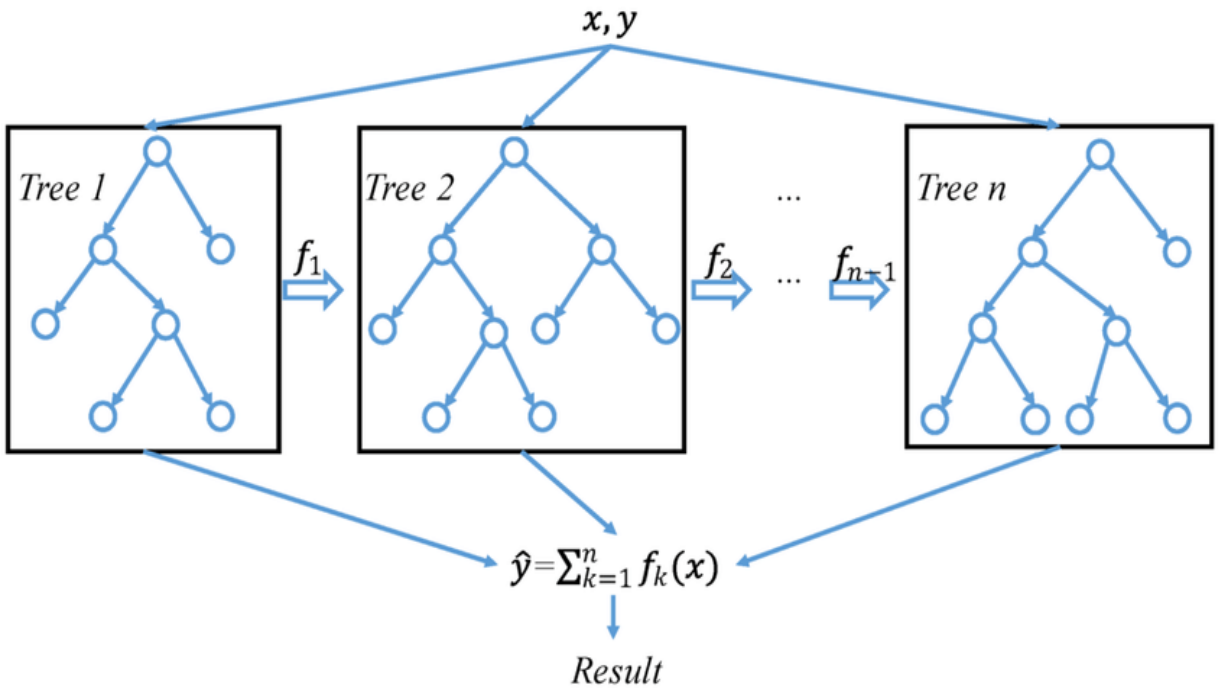

Figure 4: XGBoost architecture [17]

v) K-Nearest Neighbor

KNN is a simple and rather intuitive machine learning model that is used both for classification and regression tasks. It is a type of lazy learning or instance-based learning where it uses a majority voting system for a point to be classified into a certain class. It finds out the Euclidian distance from the point to be classified to the k number of nearest points and classifies said point by looking at the label of majority of the closest samples.

An illustration of the KNN model is presented in Figure 5:

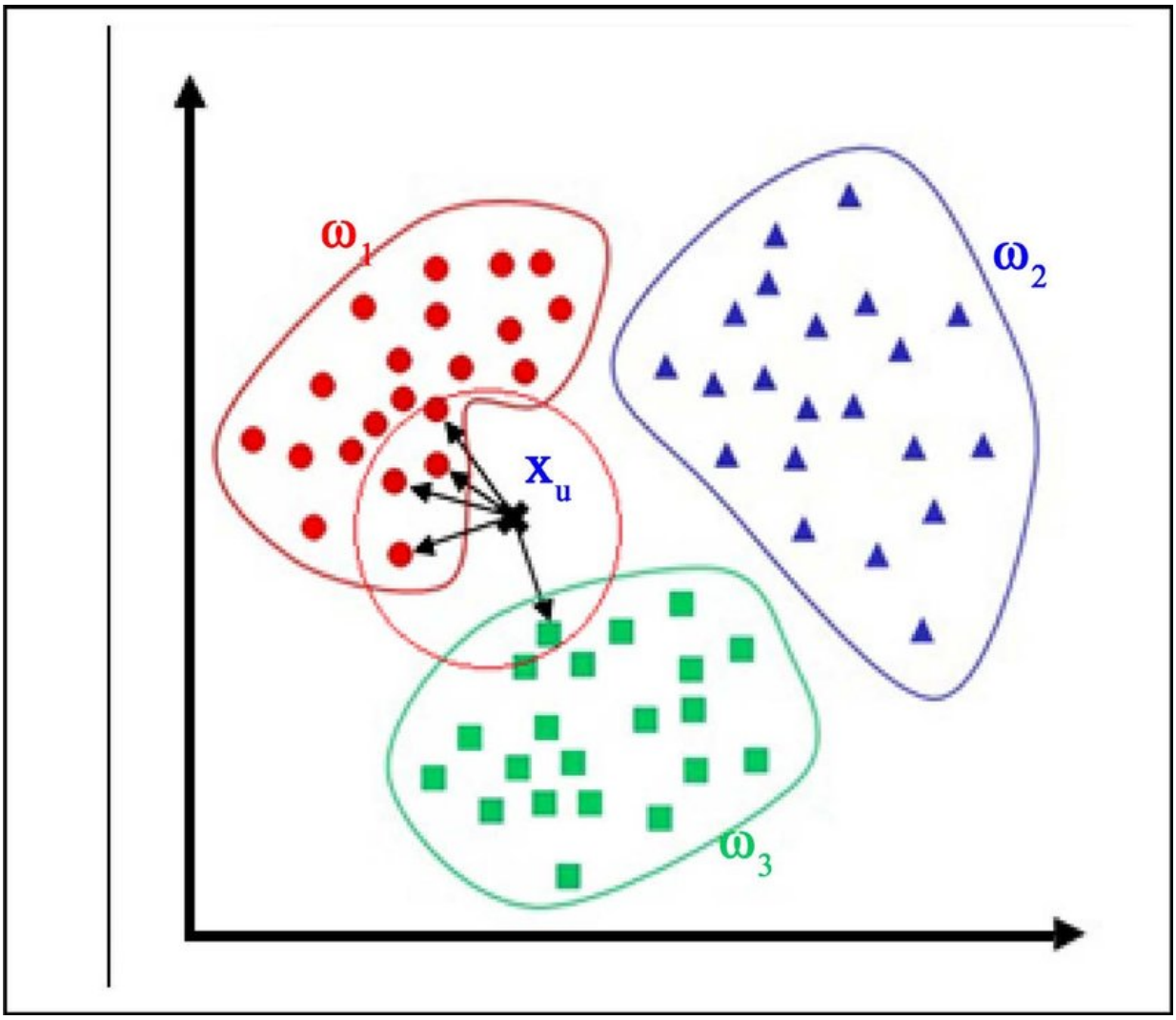

Figure 5: KNN Model [18]

vi) Artificial Neural Network

Artificial Neural Network or ANN is a type of deep neural network that is inspired by the structure of the biological neural network in the human brain. ANNs are composed of nodes and neurons which are interconnected. These allows information to be processed through weighted connections and these weights are updated and adjusted to learn the features and structures from the data. The training and adjusting of the parameters are done via backpropagation until a minimum of the loss function is reached.

The architecture consists of an input layer, hidden layers in between that does all the complex computations and the output layer that produces the final prediction. Non-linearity is introduced in the neurons by applying activation functions which improves the capability of the network to catch and learn complex patterns.

The structure of an ANN is illustrated in figure 6.

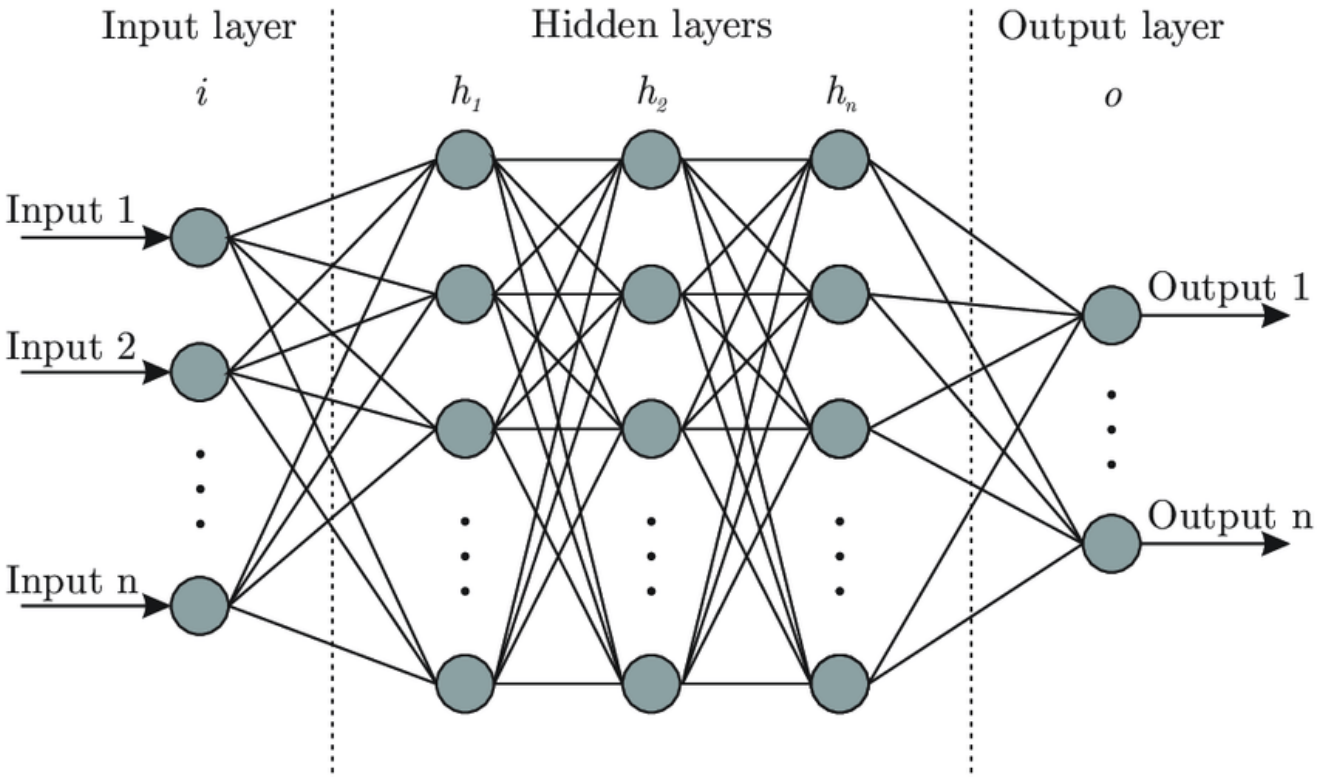

Figure 6: ANN Model [19]

vii) Voting Classifier

A very simple yet effective ensemble of machine learning models is the voting classifier. This ensemble learning model considers the decision from multiple machine learning models and aggregates it in some form to enhance the performance as well as the generalization capabilities. The voting classifier is primarily used for classification tasks, but it could be used for regression tasks as well.

There are two types of voting classifier – hard voting [20] and soft voting [21].

Hard Voting is the most simplistic type of ensemble model where the label for a specific class is chosen by aggregating all the predictions from the models being aggregated and selecting the prediction that gets the majority vote or highest occurrence from those model predictions. It works for both binary classifications as well as multi class classification.

Soft Voting classifier works by collecting probabilities, or confidence score for every class from each classifier, and aggregates them and provides the classification based on it.

6. Experimental Results and Analysis

a. Implementation Details

I first created the 2 datasets – one being the original preprocessed data and the other one with the downsampled data. I then created a test train split of 80% - with 80% of the data being used to train the model and 20% of the data as the test set.

I then ran all the models on the two datasets separately. I used GridSearchCV to find the best hyperparameters for all the machine learning models. I shall explain the implementation details of each model in details.

i) Random Forest

I created lists of parameter values for the hyperparameters - number of estimators, maximum depth, minimum sample split and minimum sample leaf.

I then used GridSearchCV from scikit-learn [29] with cross-validation of 5 with these parameters and the random forest model. I found the best hyperparameters to be no maximum depth, 1 minimum sample leaf, 2 minimum sample split and the 300 as the number of estimators.

ii) Support Vector Machine

I created the parameter list for kernels, C and gamma. I wanted to check which of the SVM kernels from linear, polynomial, rbf and sigmoid worked best for our dataset.

I then used GridSearchCV to check for the best parameters with cross validation of 5 and the SVC model. The best parameters obtained were linear kernel with C value of 10 and gamma value of 0.1.

iii) AdaBoost

I created a list of parameters with number of estimators and learning rate.

The best parameters obtained by GridSearchCV with the AdaBoost model and a cross-validation of 5 was a learning rate of 0.1 and the number of estimators to be 100.

iv) k-Nearest Neighbor

I created a list of parameters with number of neighbors, weights, and p.

The best parameters obtained by GridSearchCV with the KNN model and a cross-validation of 5 was a p value of 1, uniform weight and number of neighbors to be 9.

v) XGBoost

I created a list of parameters with learning rate, max depth, and number of estimators.

The best parameters obtained by GridSearchCV with the XGBoost model and a cross-validation of 5 was a learning rate of 0.1, a maximum depth of 7, and the number of estimators to be 100.

vi) Artificial Neural Network

I constructed an ANN model with a learning rate of 0.001 and a dropout of 30%. I added just 3 layers - the input layer with 128 neurons and ReLU as the activation function, the hidden layer with 64 neurons and ReLU as the activation function and the output layer with a single neuron and sigmoid as the activation function. I added a dropout layer after every layer.

I used the Adam optimizer and used the binary_crossentropy loss function with accuracy as the metric. I also used early stopping by monitoring the change in validation loss with a patience value of 5.

I fit the model to my data with a batch size of 32, a validation size of 20% and ran it over 100 epochs.

vii) Voting Classifier

I imported the VotingClassifier from scikit-learn and created an ensemble model using the four best performing models – SVM, XGBoost, AdaBoost and Random Forest with the best parameters from the . I implemented both hard and soft voting on it.

b. Evaluation Metrics

We used the evaluation metrics Accuracy, Precision, Recall and F1-Score to assess the performance of our model. The equations of these metrics are illustrated in equation 1 to equation 4.

$$Accuracy = \frac{t_p+t_n}{t_p+t_n+ f_p+f_n} \quad (1)$$

Here, t_p is the number of true positives, f_p the number of false positives, t_n is the number of true negatives and f_n is the number of false negatives.

Accuracy is the percentage of correctly classified samples in the dataset.

$$Precision = \frac{t_p}{t_p + f_p} \quad (2)$$

The precision is a classifier's ability to label positive samples as positive.

$$Recall = \frac{t_p}{t_p + f_n} \quad (3)$$

The recall is the classifier's ability to accurately predict the positive samples.

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (4)$$

F1-Score is the harmonic mean of precision and recall and it shows the model's capacity to ascertain a balance between precision and recall.

c. Experimental Result

The experimental results of the performance of the models RF, SVM, XGBoost, KNN, AdaBoost, ANN, and the ensembles by their metrics accuracy, precision, recall and f1-score are illustrated in the tables 1 to 4 below.

Table 1. Accuracies of the models

Models	**Accuracy on Regular Dataset**	**Accuracy on Undersampled Dataset**
Random Forest	0.86	0.86
SVM	0.86	0.87
AdaBoost	**0.87**	0.89
KNN	0.77	0.77
XGBoost	0.86	0.87
ANN	0.85	0.86
Hard Voting	**0.87**	**0.90**
Soft Voting	**0.87**	0.89

It can be noted that all the models perform pretty similarly with an accuracy of around 86-89% with KNN performing the worst with an accuracy of 77%. It can also be noted that the improvement in accuracy is quite minor for all the models with RF and KNN being unchanged. The hard voting had the best improvement in accuracy with the undersampled dataset of 3%.

On the regular dataset, AdaBoost was the best performing individual model with an accuracy of 87%. This is the same as the ensemble model's accuracy of 87%.

On the undersampled dataset, AdaBoost was still the best performing individual model with an accuracy of 89%. This is very close to the ensemble model's (hard voting) accuracy of 90% and the same as the soft voting classifier's accuracy of 89%.

Table 2. Precision of the models

Models	**Precision on Regular Dataset**		**Precision on Undersampled Dataset**	
	Class		**Class**	
	0	**1**	**0**	**1**
Random Forest	0.86	0.85	0.84	0.89
SVM	0.88	0.83	0.86	0.89
AdaBoost	0.88	**0.86**	0.86	**0.91**
KNN	0.76	0.82	0.74	0.79
XGBoost	0.88	0.81	0.84	0.90
ANN	0.86	0.81	0.86	0.86
Hard Voting	0.88	**0.86**	**0.88**	**0.91**
Soft Voting	**0.89**	0.85	0.87	**0.91**

A clear observation to be noted is there is a clear decrease in the precision in the undersampled dataset in class 0 with a maximum decrease in XGBoost of 4%, while class 1 increases between the datasets, with the maximum precision increasing by 6% in Soft Voting.

The best overall precision on the regular dataset is AdaBoost and the ensemble models. The best overall precision on the undersampled dataset is the Hard Voting classifier.

Table 3. Recall of the models

Models	**Recall on Regular Dataset**		**Recall on Undersampled Dataset**	
	Class		**Class**	
	0	**1**	**0**	**1**
Random Forest	0.93	0.73	0.88	0.85
SVM	0.92	0.77	0.88	0.87
AdaBoost	0.93	0.76	**0.91**	0.87
KNN	**0.94**	0.46	0.77	0.76
XGBoost	0.90	**0.78**	0.89	0.85
ANN	0.91	0.73	0.84	0.88

Models	Recall on Regular Dataset		Recall on Undersampled Dataset	
	Class		Class	
	0	1	0	1
Hard Voting	0.93	0.77	**0.91**	**0.89**
Soft Voting	0.92	**0.78**	**0.91**	0.88

Similar observation to precision can be noted - there is a clear decrease in the recall in the undersampled dataset in class 0 with a maximum decrease in KNN of 17%, while class 1 recall between the datasets, with the maximum recall increasing by 30% in KNN. KNN had the poorest recall for class 1 at just 46%.

The best overall recall on the regular dataset is KNN for class 0 and XGBoost and Soft Voting for class 1. The best overall recall on the undersampled dataset is the Hard Voting classifier.

Table 4. F1-score of the models

Models	F1-score on Regular Dataset		F1-score on Undersampled Dataset	
	Class		Class	
	0	1	0	1
Random Forest	0.90	0.79	0.86	0.87
SVM	0.90	0.80	0.87	0.88
AdaBoost	0.90	0.80	0.88	0.89
KNN	0.84	0.59	0.76	0.78
XGBoost	0.89	0.80	0.86	0.87
ANN	0.89	0.77	0.85	0.87
Hard Voting	**0.91**	**0.81**	**0.89**	**0.90**
Soft Voting	0.90	**0.81**	**0.89**	**0.90**

The similarities continue, with decrease in f1-score of the 0 class and increase of the 1 class. Overall, Hard Voting had the best f1-score across the dataset compared to all the models. However, Soft Voting and AdaBoost performed almost just as good. KNN had the worst f1-score overall.

d. Discussion

It can be seen that for the dataset used, the best performing model can be said to be the hard voting classifier with an ensemble using RF, SVM, AdaBoost, and XGBoost. However, the soft voting ensemble with the same models and AdaBoost performed considerably well, with being almost similar to hard voting in most metrics.

It can also be said that although undersampling did not change much to the results in terms of accuracy, it significantly improved the performance of the models in terms of precision, recall and f1-score when it came to the 1 class, showing that there was indeed some bias that was affecting the performance and it did have a considerable impact on improving the general performance of the model.

7. Conclusion

a. Summary

I set out to investigate how the different machine learning models – Random Forest, SVM, AdaBoost, KNN, XGBoost, artificial neural network model, and ensemble models performed when tasked with bladder cancer survival prediction.

We also did undersampling to remove the class imbalance issue which was affecting the performance by creating a bias. AdaBoost performed the best when it came to overall performance amongst the individual machine learning models.

The ensemble did provide a slight improvement over the regular models, but it was not significant.

b. Future Research

Further research on this domain could be conducted using deep learning models to see if the performance can be improved even further. Implementing different upsampling techniques to remove the class imbalance such as Synthetic Minority Over-sampling Techniques and others to see how that affects the performance can also be taken into consideration.

References

[1] "SEER Cancer Stat Facts: Bladder Cancer" *National Cancer Institute*. Bethesda, MD, Available: https://seer.cancer.gov/statfacts/html/urinb.html [Accessed: 02-Jan-2024]

[2] "Cancer Facts & Figures 2018". *Atlanta: American Cancer Society*, 2018.

[3] L. Breiman, "Random forests". Machine learning, 45, p.p.5-32, 2001.

[4] V. Vapnik, "The nature of statistical learning theory", Springer science & business media, 1999.

[5] Y. Freund & R. E. Schapire "A desicion-theoretic generalization of on-line learning and an application to boosting." European conference on computational learning theory, pp. 23-37, 1995.

[6] Peterson, "K-nearest neighbor." *Scholarpedia* 4, no. 2, 2009.

[7] T. Chen,"Xgboost: extreme gradient boosting" R package version 0.4-2, 1(4), pp.1-4, 2015

[8] Yegnanarayana, Bayya. "Artificial neural networks*"*, *PHI Learning Pvt. Ltd.,* 2009.

[9] Dietterich, Thomas G., "Ensemble methods in machine learning.", International workshop on multiple classifier systems, pp. 1-15. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000.

[10] Clinton et al., "Genomic heterogeneity as a barrier to precision oncology in urothelial cancer", *Cell Rep*, 2022.

[11] Z. Hasnain, "Machine learning models for predicting post-cystectomy recurrence and survival in bladder cancer patients." *PloS one*, 14(2), p.e0210976, 2019.

[12] Tsai et al. "Machine learning in prediction of bladder cancer on clinical laboratory data." *Diagnostics 12*, no. 1, 2022.

[13] N. Tokuyama, Akira Saito, Ryu Muraoka, Shuya Matsubara, Takeshi Hashimoto, Naoya Satake, Jun Matsubayashi et al., "Prediction of non-muscle invasive bladder cancer recurrence using machine learning of quantitative nuclear features.", *Modern Pathology* 35, no. 4, p.p.533-538, 2022.

[14] M. Y. Khan et al., "Automated prediction of Good Dictionary EXamples (GDEX): a comprehensive experiment with distant supervision, machine learning, and word embedding-based deep learning techniques." Complexity 2021 p.p. 1-18, 2021.

[15] "Support Vector Machine Algorithm", Javatpoint, 2023. Available: https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm [Accessed: 2-Jan-24]

[16] Wang, J., Zhang, S., Qiao, H. and Wang, J., "UMAP-DBP: an improved DNA-binding proteins prediction method based on uniform manifold approximation and projection." The Protein Journal, 40, pp. 562-575, 2021.

[17] Wang, Yuanchao, Zhichen Pan, Jianhua Zheng, Lei Qian, and Mingtao Li., "A hybrid ensemble method for pulsar candidate classification." Astrophysics and Space Science, 364, p.p.1-13, 2019.

[18] Zhang, Wenhao, "Machine learning approaches to predicting company bankruptcy.", *Journal of Financial Risk Management* 6, no. 04, 2017.

[19] Bre, Facundo, Juan M. Gimenez, and Víctor D. Fachinotti, "Prediction of wind pressure coefficients on building surfaces using artificial neural networks.", *Energy and Buildings* 158 p.p. 1429-1441, 2018.

[20] D. Ruta, and B. Gabrys. "Classifier selection for majority voting." *Information fusion 6*, no. 1, p.p. 63-81, 2005.

[21] Mitchell, H.B. and Schaefer, P.A., “A “soft” K-nearest neighbor voting scheme” *International journal of intelligent systems*, 16(4), pp.459-468, 2001.