RESEARCH

Multi-Omics analysis identifies signature genes to predict bladder cancer survival

Gia Cuong Pham, Mammadli Matanat and Hamidovic Samra

Full list of author information is available at the end of the article

Abstract

For this project, we found out our Bladder Cancer (BLCA) data, which consists of copy number variation (cnv) data, gene expression RNAseq data, survival data and phenotype data from The Cancer Genome Atlas and xenabrowser.net site, which we will cite in this report later on. Our plan/strategy was to pre-process all of these data and apply cnv and gene expression analysis with the purpose to have most bladder cancer-relevant genes simultaneously selecting features for predicting patient survival with the help of machine learning models. Besides that, we also applied survival analysis, functional annotation, and gene enrichment analysis. The primary endpoint was to predict the survival of bladder cancer patients and find out how these genes affected their life expectancy and survival and the biological mechanisms behind it.

1 Introduction

Our assigned task involved conducting a comprehensive four-part omics data analysis on a multi-omics dataset, either obtained from public repositories like ENCODE, TCGA, GEO, or through our own experiments. After thorough research and consideration of various papers and datasets, we opted to work with the Bladder cancer dataset from the TCGA database, inspired by the paper titled ”Multi-Omics analysis identifies a lncRNA-related prognostic signature to predict bladder cancer recurrence” by Zhipeng Xu et al [9]. Our primary objective was to identify a subset of genes strongly associated with bladder cancer mortality and decipher their biological significance. Initially, we explored the preprocessed dataset from the mentioned paper; however, it was limited to long non-coding RNA, which did not align with our scientific question. Consequently, we decided to proceed with the original TCGA Bladder cancer dataset to address our research question effectively. The paper mentioned served as a valuable guide, providing insights into solving our scientific inquiry. Our analysis encompassed data integration, involving quality control, biological object association, and data transformation. Subsequently, we delved into data modeling, employing statistical methods, machine learning, and visualization techniques to uncover meaningful patterns and correlations. To enrich our understanding, we also explored data annotations from publications and public knowledge bases, further enhancing the biological context of our findings. By combining these analytical steps, we aimed to unveil critical genes relevant to bladder cancer mortality and contribute to the broader knowledge in cancer research.

2 Data collection

To commence our project, we obtained the necessary datasets from various sources. The transcriptomics, survival, and phenotype data were sourced from the xenabrowser.net website, while the CNV dataset was downloaded from firebrowser.org. Our initial exploration of the downloaded datasets involved investigating potential study dropouts, identifying abnormal patterns, detecting missing values, and ensuring consistency across samples. Next, we conducted diverse analysis methods on our two primary datasets: the CNV and transcriptomic datasets. Through these analyses, we aimed to gain valuable insights into our results. By selecting a subset of genes from these analyses, we laid the foundation for employing machine learning models and conducting further investigations, such as survival analysis and gene enrichment analysis. This multi-step approach holds the potential to shed light on significant genes, unravel survival patterns, and contribute to the advancement of our understanding of cancer research.

Our primary drive for undertaking this project stemmed from our desire to predict the survival of bladder cancer patients, a prevalent and challenging cancer type. Our focus was on leveraging omics data analysis to identify key genes responsible for patient outcomes. As we delved into the wealth of research and studies on survival analysis conducted worldwide, we became captivated by its essential role in targeting aggressive cancer types, devising effective therapies, and advancing anti-cancer drug development. Our goal was to deepen our understanding of these methodologies, develop a robust model, and conduct similar analyses independently, thereby contributing to global efforts in cancer survival research.

3 Methods

In this part, we will briefly describe our analysis methods. Figure 1 shows us an overview of the analysis methods, that we have used in this project.

3.1 CNV analysis

Our initial approach involved performing CNV analysis using data obtained from the firebrowser.net website. The dataset consists of 412 samples, with each sample containing copy number variants from both normal and tumor cell types. To distinguish between normal and tumor cell types in the 412 samples, we relied on the information provided by the TCGA website [6]. The data for each CNV included information such as chromosome number, start and end positions of the CNV, probe value, and segmean value. The probe value in the CNV data indicates the number of short, single-stranded DNA molecules specifically designed to bind to a complementary target sequence. Higher probe values may suggest a higher copy number, while lower signal intensities could indicate a deletion or lower copy number in that particular genomic region. The segmean value, on the other hand, represents the signal intensities from multiple probes, and it helps in identifying abrupt changes in copy number, which are used to define the boundaries of different segments. Each segment corresponds to a specific genomic region with a consistent copy number state, which could be a deletion, duplication, or normal diploid copy number.

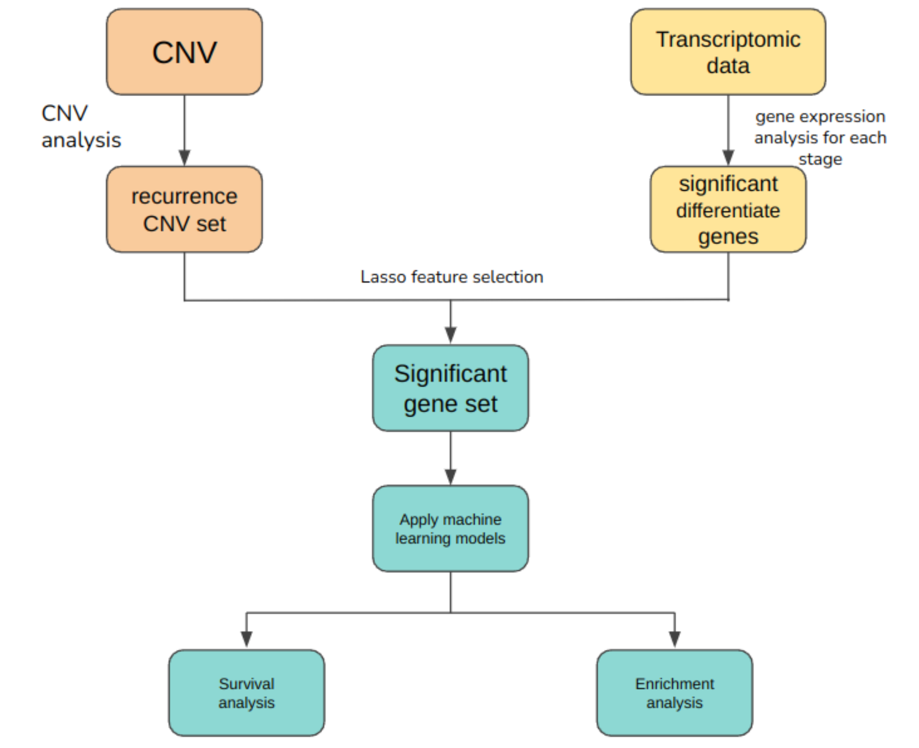


Figure 1: Overview of project's pipeline

For the CNV analysis, we aim to find independent and recurrent copy number abbreviations. For that, we used an R package called GAIA (Genomic analysis of significant chromosomal abbreviations) [5]. To be able to use GAIA, we first need to assign loss and gain CNV based on the segmentation mean value of each CNV. The used criteria value is 0.3, which means, every CNV, which has a segmentation mean value greater than 0.3, is assigned as a gain CNV, and every CNV, which has a segmentation mean value smaller than 0.3, is assigned as a loss CNV. Afterward, the dataset is ready to apply to GAIA. GAIA can be summarized as a procedure containing two main steps: Significance testing and Homogeneous peel-off.

Statistical significance testing: This step involves assessing the statistical significance of observed genomic aberrations across different samples at a specific genomic site. The null hypothesis assumes that the genomic locus is not a site of recurrent Copy Number Aberration (CNA). To estimate the distribution of test statistics under the null hypothesis (null distribution), random permutations are performed.

Homogeneous peel-off: The peel-off is an iterative procedure designed to identify significant peaks within a genomic region. In this process, selected peaks are iteratively removed based on their significance levels, and the remaining significance values are corrected using the False Discovery Rate (FDR) method (Storey et al., 2004). The peel-off procedure continues until no further peaks above the significance threshold are detected. Here, we propose an innovative peel-off approach that considers both the statistical significance and the homogeneity within samples to refine the identification of significant peaks.

Afterward, we received a list of CNVs with an adjusted p-value for each CNV segment. The significant recurrent CNVs are the CNVs that have adjusted p-value smaller than 0.01.

3.2 Gene expression analysis

For the gene expression analysis, we used the transcriptomic data, consisting of RNA-Seq data in the htseq format, and applied DESeq2 on each stage to compare them with each other. We extracted the raw counts by converting the htseq values to raw counts with the formula $\log_2(\text{raw\_count} + 1)$ so that we can use the raw counts to apply our differential gene expression analysis.

DESeq2 is a statistical tool to identify differentially expressed genes from RNA-Seq data. It takes into account the inherent variability in the sequencing data and corrects for it, providing more accurate results. The tool was developed as an improvement over the original DESeq package and has since become a widely adopted tool for analyzing RNA-Seq data.

Firstly, we plotted the distribution of our transcriptomic data (Figure 10). In figure 11 we tried to improve the result from before by truncating the counts, so we can see that most genes have a low number of counts. We tried to improve this result as well by doing a log2 transformation of the reads (Figure 12).
Afterwards, we created histograms for each stage (excluding stage 1). In figure 13, the distribution of the adjusted p-value from the genes for all stages. The adjusted p-value is on the x-axe.
Furthermore we created a density plot (Figure 14), which shows blue and red curves. The blue curves stand for survival and the red curves for non-survival. The x-axe stands for the number of reads. In the next step we created MA-plots (Figure 15) and volcano plots (Figure 16) again for each stage. The MA-plots show the mean of the normalized counts on the x-axe and the log fold change on the y-axe. The blue color represents the different expressed genes. For the volcano plots, we have chosen a dash line with 1.3 (parallel to the x-axe) and a absolute log fold change greater than 1 (parallel to the y-axe). Also we used a adjusted p-value threshold of 0.05. On the left side are the downregulated genes and on the right side the upregulated genes. All the genes smaller than the dash line are not significant.
Lastly we created the plot in figure 17 to show the number raw counts.

3.3 Multi-Omics integration

The set of significant recurrent CNVs was then annotated to genes by using biomart [1] and GenomicRanges R packages [3]. After that, the gene set, which contains CNVs, was merged with the significantly different expressed gene set from doing gene expression analysis at each stage. Simultaneously, with the purpose of finding genes, which have surely impact on the phenotype, and also reducing the number of features, which are later applied to the machine learning model, we utilized lasso regression for feature selection from glmnet package [2].

Lasso regression is an extension of linear regression that introduces a penalty term based on the L1 norm of the model's coefficients. By adding the L1 norm penalty, it encourages sparsity in the coefficient vector and drives some coefficients to exactly zero, effectively performing feature selection. In this project, we chose a penalty value of 0.01, since we don't want to lose so many features and data information. As a result, only a subset of features remains in the final model, making it more interpretable and potentially reducing overfitting.

3.4 Data preprocessing

After applying features selection with lasso regression, we then gathered the data from the raw read count data set with selected genes (features).In total, the dataset has 292 samples, and each sample has 242 genes (features). This dataset is then split to train and test set with a ratio of 3:2.

Besides the key step feature selection for improving the model's performance, we also utilized the oversampling method on the training set, since we realized that there was an imbalance between two classes of non-survival with 62 patients and survival with 113 patients. To do that, we utilized SMOTE method. SMOTE works by generating synthetic samples for the minority class, increasing its representation in the dataset. It creates synthetic samples by interpolating between existing minority class instances, effectively generating new data points in the feature space.

3.5 Applying Machine Learning models

In this project, we applied three machine learning models: random forest, XGBoost, and ridge regression.

Random Forest is an ensemble learning method used for both classification and regression tasks. It creates multiple decision tree models during the training process and combines their predictions to make more accurate and robust predictions. To start, Random Forest creates many small decision trees, and for each tree in the Random Forest, a decision tree is trained on the bootstrap sample. At each node of the tree, the algorithm selects the best feature to split the data based on some criterion, typically the Gini impurity (for classification) or mean squared error (for regression). At each node of the decision tree, only a random subset of features is considered for splitting. Suppose there are M features in the dataset, and m ¡¡ M is the number of features considered at each node. The value of m is typically set as the square root of M for classification tasks and the logarithm base 2 of M for regression tasks. Once all the trees are trained, when making predictions, each tree in the Random Forest produces an output (class label for classification or numerical value for regression). The final prediction is determined by aggregating the predictions from all the trees. For classification tasks, the mode (most frequent class) of the class labels is taken.

Similar to Random Forest, the XGBoost classifier is a scalable machine-learning system for tree boosting. XGBoost iteratively builds an ensemble of decision trees by correcting the mistakes made by previous trees. It combines weak learners, also called decision stumps, and evaluates their performance using a loss function. The loss function measures the discrepancy between the predicted and actual labels. Equation (1) shows us the formula to build the next tree from the previous tree in the XGBoost algorithm. As it is shown, by adding the loss function from the previous tree multiplied by a learning rate to create the next tree. This iterative approach results in an ensemble of boosting trees.

$$F(m) = F(m-1) + \mu * -\frac{\partial(L)}{\partial F(m-1)} \tag{1}$$

where F(m-1) stands for m-1 tree μ is learning rate.
After having an ensemble boosting trees model, equation (2) has been used to predict the output based on a given dataset, where $\hat{y}$ stands for predicted output,$\gamma_t(x)$ is the prediction made by an individual tree t for input x, λ is the regularization term that penalizes the complexity of the model and $f_t(x)$ is the complexity of the tree t for input x.

$$\hat{y} = \sum_{t=1}^{T} \gamma_t(x) + \lambda \sum_{t=1}^{T} f_t(x) \tag{2}$$

Ridge classification is a variant of the Ridge regression technique, which is commonly used in machine learning for regression problems. However, instead of regression, Ridge classification is used for binary or multi-class classification tasks. In traditional Ridge regression, the objective is to fit a linear model to the data by minimizing the sum of squared errors between the predicted values and the actual values, while also adding a penalty term to the regression coefficients. The penalty term is proportional to the square of the L2 norm (Euclidean norm) of the coefficients. This penalty helps to regularize the model, preventing overfitting and reducing the impact of multicollinearity (high correlation between features).

For tunning the hyperparameters, we utilized the gridsearchCV function of sklearn, in which gridsearchCV probes every combination of the given hyperparameters and give us the best set of hyperparameters based on our model evaluation method.

3.6 Model evaluation & feature importance

We conducted a thorough performance assessment of the model using several evaluation techniques. The first step involved utilizing the evaluation metrics, including precision, recall, and f1-score, which was generated using the "classification_report" function from the scikit-learn library. The resulting matrix provided valuable insights into the model's performance by presenting those values.

In addition to the evaluation metrics, we also utilized the ROC curve to evaluate the model's performance and assess any potential overfitting issues. The ROC curve is a graphical representation that illustrates the trade-off between sensitivity (true positive rate) and specificity (1 - false positive rate) at different classification thresholds. This visualization enabled us to understand the model's discriminatory power and its ability to make accurate predictions across various threshold levels.

To ensure the reliability of our results, we employed cross-validation to plot the confidence interval of the Area Under the Curve (AUC) metric. This approach helped us gain more confidence in the AUC values and assess the model's generalization ability to unseen data.

Furthermore, we performed feature importance analysis using the built-in "feature_importances" attribute of the Random Forest model, since this model gave us the best result in comparison to the others. This analysis allowed us to identify which features had the most significant impact on the model's predictions. Subsequently, we retrieved the important features from the preprocessed dataset to gather further information about their contribution to the model's performance.

By combining these evaluation techniques, we obtained a comprehensive understanding of the Random Forest model's performance, its ability to generalize to new data, and the critical features that influenced its predictions. This comprehensive evaluation provides valuable insights and ensures that we have a reliable and accurate model for our specific task.

3.7 Survival analysis

With our phenotype dataset, we edited four stages of bladder cancer in our survival dataset. The phenotype dataset was filled with 422 samples of the tumor diagnose stage in the beginning. Afterwards, we had to throw 150 samples.
We figured out that the stage might be a confounder, so we took a closer look at the four stages individually. Our first attempt was to make a survival analysis on the four stages, for a better prediction. For example if a patient has stage 4 cancer, we can look only at stage 4 and not on all the stages combined, to predict the chance of survival for this patient. In figure 18 there are four kaplan-meyer curves pictured, one stands for a stage. What stands out is stage one. For our further analysis, we excluded stage one due to the very low sample size and no patient died in this stage. We can see the in the y-axe the survival rate or overly survival (OS) in dependence of the overly survival time (OS time). In the next steps we looked at each stage and visualized the survival rate of the patients in each stage in the figures 19, 20 and 21 for better results.

For the next step, we used the transcriptomic data and the survival data with the included stages. To create figure 18 we only needed our survival dataset and a simple function in R. The transcriptomic data was needed to create the plots in figure 26 till 37. From the feature importance part, we took the first four genes (Table 2) which are in our transcriptomic data. Firstly, we made survival datasets for each stage. Next up, we added the median of the read counts for each gene in our transcriptomic data. After that, we created four tables of the transcriptomic data, a table stands for one gene and in our case the genes where DIXDC1, DPYSL3, TMEM74 and YAP1 as mentioned from the feature importance part. In the tables, we included a strata with the labelling "HIGH" and "LOW". If a single read count from the gene is smaller than the median, it will get the label ”LOW” and if its higher than the median it will be labeled as ”HIGH”. The last step was merging each of these four tables with our survival data by sample. With the finished tables, we were able to depict each gene for all stages and for each stage. To be exact, we depicted the low read counts and the high read counts of a gene as a kaplan meyer curve.

3.8 GO enrichment analysis

Lastly, we used GO enrichment analysis on significant genes acquired from previous analysis methods. Gene Ontology Enrichment analysis is used to interpret high-dimensional molecular data and generate hypotheses about biological mechanisms/phenomena behind specific functions and experiments. Main goal of this type of analysis is to find out up or down-regulated, over and under-represented genes and proteins in a given gene/protein set and using annotations for the given gene/protein set.

First, we read our significant genes and then pointed out their ENSEMBL IDs as a separate column. We used these IDs for GO enrichment analysis with the help of enrichGO() function, as ontology method we used Molecular Function, Biological Process and Cellular Component. We plotted all of these 3 GO analysis results with the help of Barplots and got our top/highest count results for each section (see plots under Results section).

Then we used clusterProfiler library’s compareCluster() function with Molecular Function as ontology method, to visualize these genes as clusters that they build with similar molecular functions. First we got our dotplot (again, from clusterProfiler library) of molecular functions of these significant genes from specific biotypes. Then we used emapplot() function from enrichplot package, to get our clusters and plotted these clusters with the help of cowplot package’s plot_grid() function. The result is seen below in Figure 9.

4 Results

In this part, we will briefly describe the result of each part of the project.

4.1 CNV analysis

After applying GAIA with the CNV dataset, we received a file called Tumor.All.txt, which contains CNV segments across all samples. Besides CNV segments, it also provided us CNV abbreviation, length of CNV and adjusted p-value for each CNV. From Figure 10, as we can see, our adjusted p-value distribution skews on the left side, and starting from an adjusted p-value of 0.01, there are not many CNV segments. It could be that our data was somewhat imbalanced between the number of normal and tumor cell types. Hence we chose an adjusted p-value threshold of 0.01 with the purpose of reducing as much as possible the false significant cnv segments. With that threshold, we gathered 3448 significant CNV segments across 24 chromosomes. With the help of GenomicRange and biomart, we were able to find 11591 genes, which lay in between significant CNV segments. After that, we cross-checked with the genes profile of gene expression analysis, we had 9254 genes left.

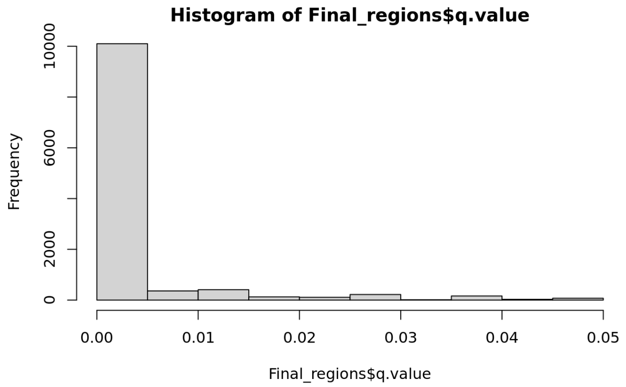


Figure 2: adjusted p-value distribution of CNV segments

4.2 Gene expression analysis

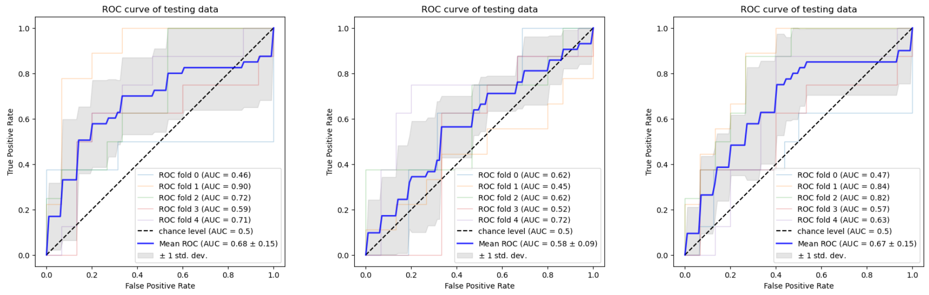
From our analysis we got 619 significant genes and 111 samples in stage 2, 1003 significant genes and 97 samples in stage 3 and 607 significant genes and 84 samples in stage 4.

In figure 11 its clear that the majority of genes have a low read count due to the large first bar. We did a log2 transformation on the counts per gene (Figure 12) to get a better result. Unfortunately it looks quite similar to the plot before. As for the histograms there are only stage 2 and 3 which have a even distribution (Figures 13a, 13b). On the contrary, the p-value distribution in stage 4 is quite insignificant (Figure 13c). Thats because the majority of genes in stage 4 shift to the right side, where the adjusted p-value is large. Most of the genes have a p-value greater than 0.9 which is not significant. One crucial reason for that is the small sample size we had for stage 4. So based on this and the results in figure 13c we lack of evidential proof that there are actually significant genes. We also may have excluded some significant genes unintentionally so some genes got lost in the process.

4.3 Applying machine learning models

As mentioned above, the hyperparameters of each model are tunned by using gridsearchCV with 5 folds cross-validation on the train set. Each model is then assigned the best hyperparameter for label prediction. Starting with random forest, gridsearchCV gave us the random forest model that has a maximum depth of 10, criterion gini, and max features log2. With ridge regression, the best hyperparameter was the alpha value of 0.01, and with XGBoost, the best hyperparameter set was the learning rate value of 0.01, and the gamma value of 0.01.

Figure 3 shows us the ROC curve of each model on the test set with 5-fold cross-validation. As we can see, the mean AUC value of the random forest model is the highest at 0.68. Following that, it is XGBoost with a mean AUC value of 0.67, and the lowest AUC value of 0.58 belongs to the ridge regression model. Besides that, all ROC curves fall within the AUC confidence interval, indicating reasonably reliable results. Hence, we can conclude that Random forest surpassed the ridge regression



(a) Random forest (b) Ridge regression (c) XGBoost

Figure 3: ROC curves of three machine learning models

model and the XGBoost model. However, the AUC value is still low, and we can also see that from the t-SNE plot in the appendix, since the data points of survival and non-survival mimic each other. Besides that table 1 shows us also the precision, recall, and f1-score of each model. With that metrics, we realized that our training step is overfitting because all the metrics have the same result in all models. Our assumption is that, because the number of features is large than the sample size and besides that, most of our non-survival samples in the training set are simulated. However, we still chose the Random forest model to do further analysis, since it gave us the best AUC value.

Table 1: Test classification performances: Sensitivity, Precision and F1 score for Random forest, ridge regression and XGBoost on each class

	Precision			Recall			F1-score		
	RF	Ridge	XGBoost	RF	Ridge	XGBoost	RF	Ridge	XGBoost
survival	0.74	0.74	0.74	0.84	0.84	0.84	0.79	0.79	0.79
non-survival	0.60	0.60	60	0.44	0.44	0.44	0.51	0.51	0.51

4.4 Features importance

Using the built-in function ”feature_importances” of the sklearn library, we obtained the Gini impurity values for each feature calculated during the tree-building process. Figure 4 displays the Gini impurity values for the first 18 most important features. The x-axis represents the Ensemble IDs of the features, and the y-axis shows the corresponding Gini impurity values.

To identify the gene names associated with each Ensemble ID, we cross-referenced the IDs using the genecards.org website. Table 2 presents the Ensemble ID in the first column, the corresponding gene name in the second column, and the role of each gene in the cancer-development process in the last column.

According to the research findings from the cited papers, several genes identified by their Ensemble IDs have important roles in bladder cancer development and patient outcomes. For example, the DIXDC1 gene was found to promote G0/G1 to S phase transition, leading to increased cyclin D1 expression and decreased p21 protein expression, both of which are factors contributing to tumor growth in bladder cancer [7]. DPYSL3 gene overexpression was associated with bladder cancer recurrence and correlated with tumor aggressiveness and poor patient survival [4]. The TMEM74 gene's overexpression was linked to shorter survival periods in various cancer types [10]. YAP1 gene was significantly associated with the development and metastasis of human bladder cancer [8].

The cumulative evidence from these studies suggests that these genes, directly or indirectly, influence the mortality of bladder cancer patients. Their roles in various aspects of cancer development make them relevant targets for further research and potential therapeutic interventions.

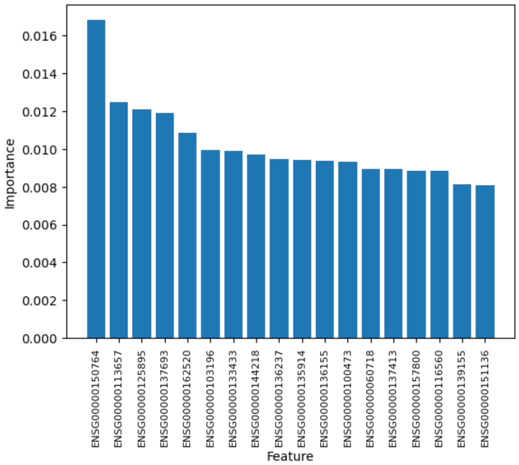


Figure 4: Feature importance of first 18 features with Random forest model

Table 2: Cancer-Related Genes

Ensemble ID	Gene Name	Cancer-Related Information
ENSG00000150764	DIXDC1	Participating in growing of tumor
ENSG00000113657	DPYSL3	High DPYSL3 expression predicted higher bladder tumor recurrence rate
ENSG00000125895	TMEM74	High expression of TMEM74 shortens the surviving periods of patients in several types of cancer
ENSG00000137693	YAP1	YAP1 plays an important role in the development of bladder and is significantly associated with the development and metastasis of human bladder cancer

4.5 Survival analysis

The first thing we can state, is that in stage four more patients die after a short amount of time in comparison to the other stages (Figure 18).

But as we mentioned in the beginning, the cancer stage is a confounder due to its low sample size. That becomes more clear in the following figures. The barplots for each stage show how much patients survived (dark blue or 0) and how much patients died (light blue or 1) and in what time frame. In figure 19 most patients survive, but there are still a few who died after a very short time (light blue). These patients probably did not die, but dropped out. Another good example for this confounder can be seen in figure 21. The distribution of survival and non-survival patients is quite equal, but that could be due to the small sample size of stage 4. And another unusual aspect is that in stage 4 there are a few patients who die after a long time, not like the majority who die after a short amount of time. Although we cannot do anything about these few confounders in our data, we still have to keep them in mind but also continue with our analysis.

The last part of the survival analysis consists of depicting the survival rate of the four genes for each gene as a high expressed and low expressed gene. In the plots are two kaplan meyer curves for each gene. The blue one shows the survival curve if the gene is downregulated (low read counts) and the red one if the gene is upregulated Let's take a look at the DPYSL3 gene first. As we know from the feature importance part a high expression of DPYSL3 predicted a higher bladder tumor recurrence rate in patients. With our analysis, we wanted to confirm the information from the paper with our results. Figure 26 confirms the information, as you can see that the high regulated curve drops faster, so more patients died. After we looked at all stages, we depicted the gene for each stage (Figure 27, 28, 29). Stage 2 is not exactly what we wanted to achieve (Figure 27), but in the other stages its similar to the information from the paper.

A good example for how it didn't work as we expected, is the visualization of the gene TMEM74. In the paper it says that a high expression of TMEM74 significantly shortens the surviving periods of patients in several types of cancer. Contradictory to this fact is our plot in figure 30. However if TMEM74 is downregulated, it falls rapidly, which means more patients die if this gene is downregulated. So our result shows the complete opposite of what is written in the paper about this gene. We plotted the down and upregulated gene for each stage, but we came to the same false result.

So when we compare our results with the information from the paper, our data gave us a different solution.

4.6 GO Enrichment Analysis

As a result of GO Enrichment analysis, we got our Barplots from different ontology methods. Adjusted p values are colour-coded and the lower the p value is, the more likely is that Null hypothesis is false and should be rejected, and the more likely it is, that our results are statistically significant. Lower adjusted p values have reddish colour, higher adjusted p values have purple colour.

When we used as an onthology method ”Molecular Function”, we got two highest count of genes with functions such as tubulin binding and metal ion transmembrane transporter activity (Figure 5). We further investigated these functions and their role in bladder cancer progression and this is what we found out:

Tubulin is a key component of microtubuli, which are responsible for maintaining cell shape and play an important role in cell division. Therefore higher tubulin binding activity means anti-cancer drugs can easily interact with tubulin and result in inhibition of cell division.

Metal ion transmembrane transporter activity had also the highest count of our significant genes. Transmembrane transporter enables transfer of specific substances from one side of membrane to another. Therefore metal ion transmembrane transporter enables transfer of metal ions. After searching it up, we found that in Bladder cancer, cation channel activity alterations are reported. For ex:

In Bladder cancer cells, Voltage-gated sodium channels were upregulated, which can be linked to tumor aggressiveness.

Calcium-channel activity was are also upregulated, which we all know, Ca is involved in cell proliferation and migration.

But Potassium channels were downregulated, which also results in higher regulation of cell proliferation, migration and invasion.

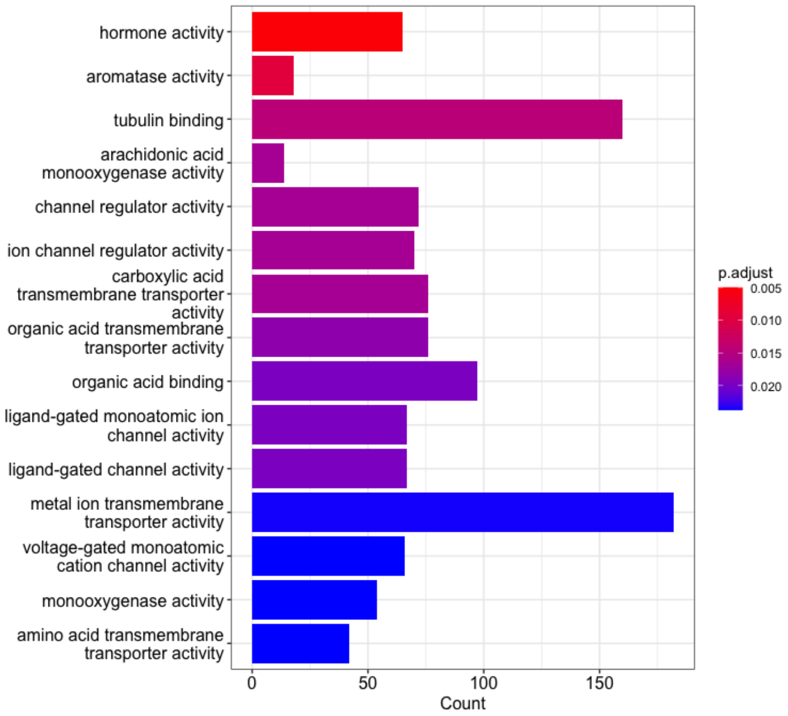


Figure 5: GO Enrichment analysis of our significant genes, Molecular Function.

Using Biological Process as our onthology method, we got highest counts of genes with response to xenobiotic stimulus and muscle contraction (see Figure 6).

Cellular response to xenobiotic stimulus means any process that results in a change in state or activity of a cell or an organism (in terms of movement, secretion, enzyme production, gene expression, etc.) as a result of a stimulus from a xenobiotic, a compound foreign to the organism exposed to it. It may be synthesized by another organism (like ampicilin) or it can be a synthetic chemical. And after searching for this topic and it's association with bladder cancer, we found out that xenobiotic activity increases in muscle-invasive bladder cancers. [**?**]

Also bladder cancers are often grouped based on if they have invaded into the main muscle layer of the bladder wall. A bladder cancer that has not grown into the muscle layer can be described as superficial or non-muscle invasive bladder cancer. Non-invasive bladder cancer cells show high activity in muscle contraction.

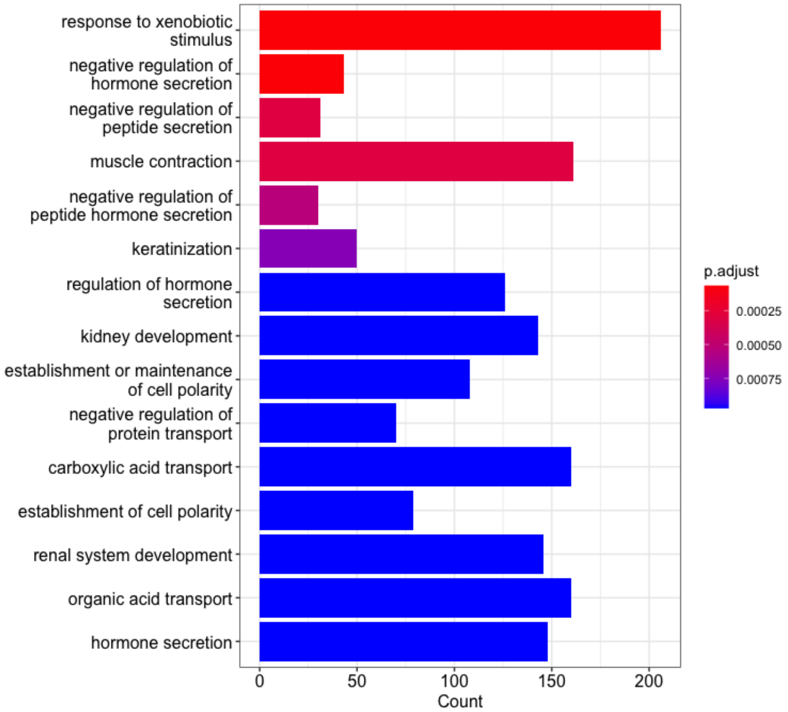


Figure 6: GO Enrichment analysis of our significant genes, Biological Process.

When we used Cellular Component as our onthology method (Figure 7, we noticed the highest counts of genes in collagen-containing extracellular matrix and in apical part of cell, as seen in Figure 7.

The extracellular matrix (ECM) plays a key role in the modulation of cancer cell invasion. In Bladder cancer the role of ECM proteins has been widely studied. The mechanisms, which are involved in the development of invasion, progression and generalization, are complex, depending on the interaction of ECM proteins, especially structural proteins such as collagens with each other as well as with cancer cells. [**?**]

Apical layer - The innermost layer is a barrier between the bladder lumen and the underlying tissue. It is a single layer of umbrella-shaped cells (i.e., umbrella cells) that are frequently binucleated. These apical umbrella cells of the urothelium form an impermeable barrier. In bladder cancer urothelial cells, apical plasma membrane shapes into microvilli, uroplakins on the apical surface, and establishes tight junctions.

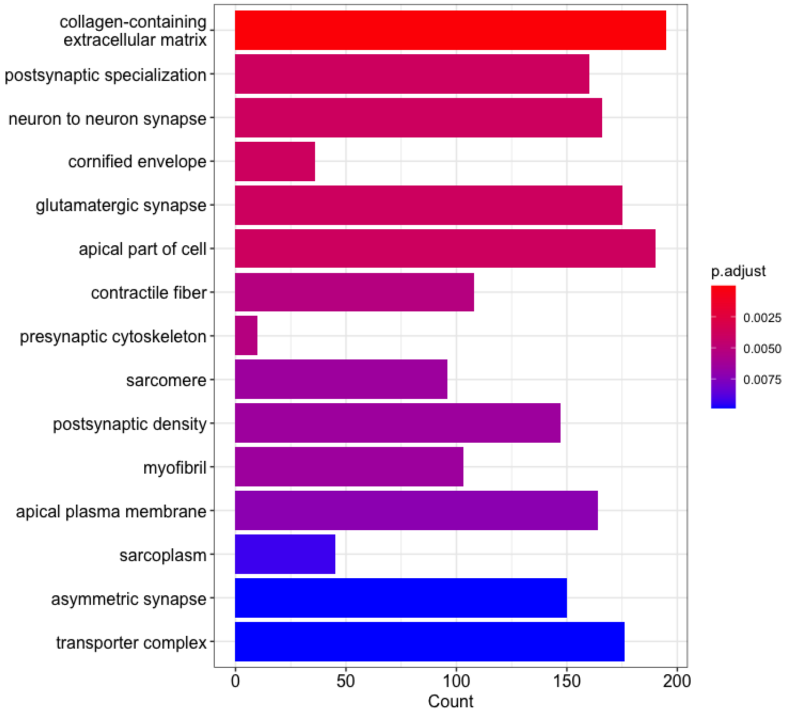


Figure 7: GO Enrichment analysis of our significant genes, Cellular Component.

We also got the dotplot of our significant genes (their gene biotypes on x Axis) and their molecular functions (on y Axis), see Figure 8.
We also have different Gene Ratios in this plot, the bigger the circle, the higher ratio we have, which means most (GeneRatio > 0.50) or all (GeneRatio = 1.00) genes from that biotype had this specific molecular function shown on y Axis.
For ex, in our plot antigen binding activity from IG V gene biotype, translation repressor activity from miRNA biotype, snRNA binding function from snRNA biotype etc. had highest Ratio (1.00) and significance (red colour).

At last, we got our compareCluster and emap plots, which we visualized them as cowplot in final form (Figure 9), which were describing our significant gene clusters. Here we can observe, that genes that are located close to each other on DNA and share a generalised function, build clusters together. For ex, in cluster close-ups we can see, that peptide binding gene, MHC protein binding gene and peptide antigen binding gene build a cluster together, as well ligand-gated channel activity gene, gated channel activity gene, monoatomic ion gated channel activity gene and voltage-gated monoatomic cation channel activity genes build a cluster together.

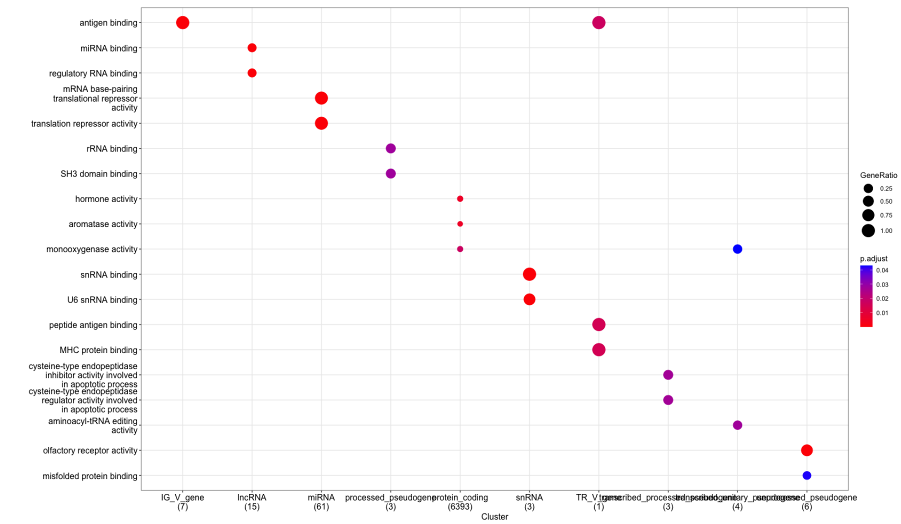


Figure 8: Dotplot of Molecular Function of significant genes from specific biotypes.

All those genes share similar molecular function.

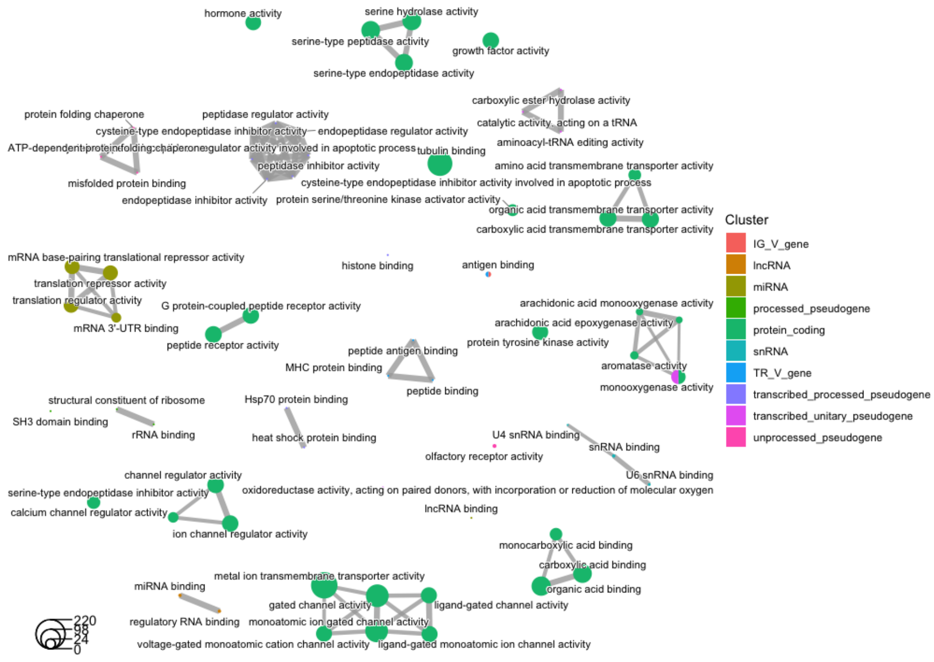


Figure 9: Clusterplot of Molecular Function of significant genes from specific biotypes.

5 Discussion

After finding out the data and preprocessing it, we practiced copy number variation analysis and gene expression analysis on our datasets, to get significant genes. Our end goal was using cnv data and transcriptomic data to predict the survival of Bladder Cancer patients and finding key genes that contain cnv and play an essential role in the survival of patients. We also used functional annotation on those genes. With cnv analysis, our aim was to find recurrent and independent copy number abbreviations. This filtered and gave us the final number (9254) of significant genes. With gene expression analysis we used DESeq2 method. As mentioned above, DESeq2 is a statistical tool to identify differentially expressed genes from RNA-Seq data. While doing gene expression analysis, we also compared different stages of cancer.

We also applied three machine learning methods to our data, Random forest, XGBoost, and Ridge regression, to predict the accuracy of our model, improve our model and prevent overfitting. Random forest showed the best results out of all three methods. We also did hyperparameter tuning for random forest and got results such as a random forest model that has a maximum depth of 10, max features of log2, and criterion gini. But unfortunately, our model was still overfitted.

Later on, we performed feature importance analysis on the Random forest model, to find out the most important and significant features that had most contribution to the model's performance. Through this evaluation technique, we obtained a comprehensive understanding of the Random Forest model's performance, its ability to generalize to new data, and the critical features that influenced its predictions. The genes that we noted for further analysis were: DIXDC1, DPYSL3, TMEM74, and YAP1.

We also did survival analysis on our phenotype dataset. We did the analysis on all four different stages, to find out the differences between stages and predict the chance of survival in each stage. Later we used the Kaplan–Meier estimator (also known as the product limit estimator) to estimate the survival function from lifetime data. In research, it is often used to measure the fraction of patients living for a certain amount of time after treatment. We got Kaplan-Meier plots of all four stages. The stage was our confounder. For further analysis, we excluded stage one, because of the low sample size and no death occurring in this stage.

Afterward, we also included genes that we took from feature importance, did survival analysis, and got a Kaplan-Meier plot for each gene. This way we were able to depict each gene for all stages and for each stage. We could find out, if up- or down-regulation of these genes influenced the survival time of patients in each stage, based on high or low read counts of a gene in a Kaplan-Meier curve.

Finally, we used GO Enrichment analysis on significant genes that we got from previous analysis methods. This Enrichment analysis interprets high-dimensional multi-omics data and gives us info about functions or biological mechanisms behind each gene or protein in our dataset. It finds out up- or down-regulated genes or proteins in a given gene or protein set and applies functional annotation (attaches biological information) on these genes.

We plotted our results from GO Enrichment Analysis and got a significant amount of information about the roles of these up- or down-regulated genes in bladder cancer survival and spreading of it. We searched for papers that gave us insight into how up- or down-regulation of these genes impact bladder cancer, how they are correlated etc. We also plotted adjusted p values of these genes to make sure that our predictions of their functions and locations were correct (lower p values). We also got clusters of our genes and could observe, that genes/proteins with similar functions were building clusters.

Overall, we learned a lot working with our dataset and doing all these important and widely used analysis steps. We feel that we now understand Data analysis, visualization, and study predictions and results better and we can contribute to cancer research and data model improvement/evaluation with all the knowledge that we gathered. We also noticed that we can work well as a team and share responsibilities, which are all useful skills for the future. We are happy and proud of our results.

6 Contributions

1 Pham Gia Cuong: Writing introduction, Data collection, method and result of cnv analysis, method and result of applying machine learning models.

2 Samra Hamidovic: Writing abstract, method and result of gene expression analysis, method and result of survival analysis.

3 Matanat Mammadli: Writing abstract and introduction, method and result of GO Enrichment analysis and discussion.

4 ChatGPT: Rephrasing text in the introduction.

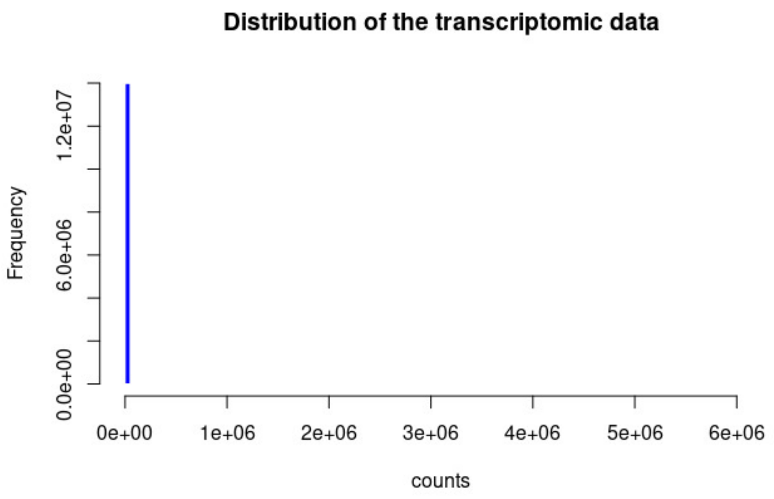


Figure 10: Distribution of Transcriptomic data

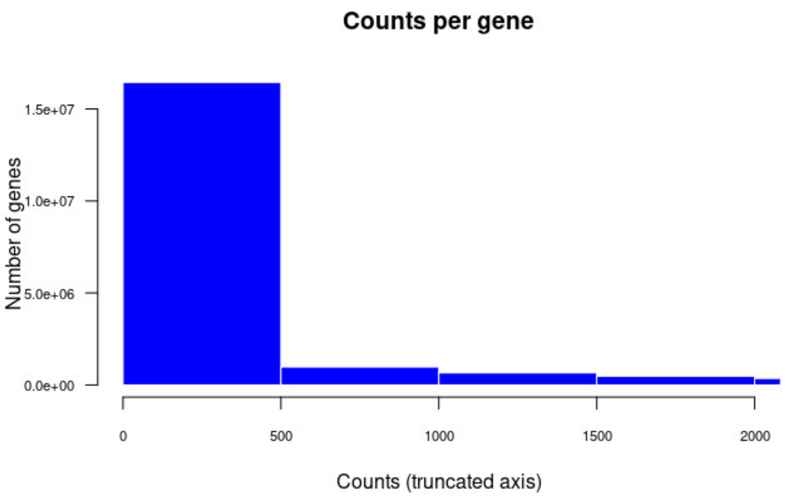


Figure 11: Counts per gene

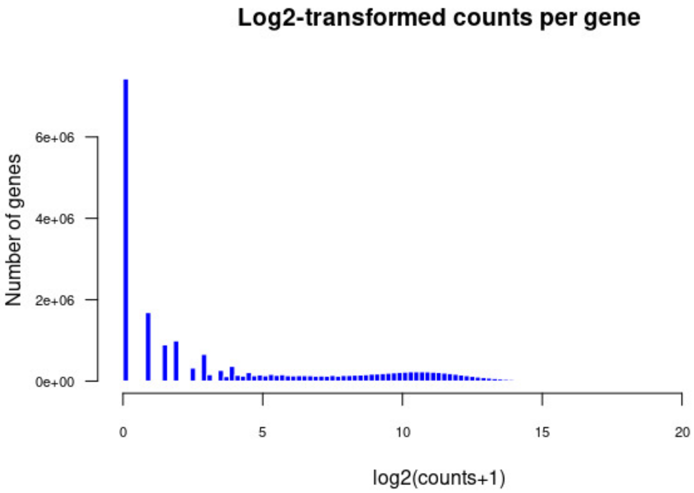
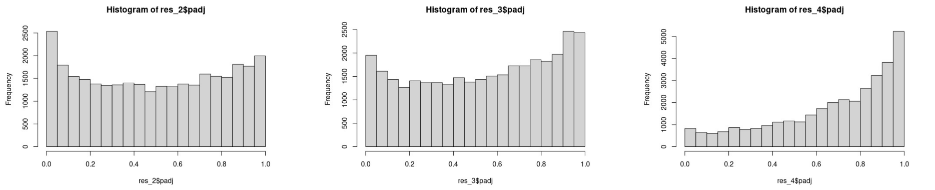


Figure 12: Log2 transformed counts per gene



(a) Histogram of second stage cancer

(b) Histogram of third stage cancer

(c) Histogram of fourth stage cancer

Figure 13: Histogram of all cancer stages

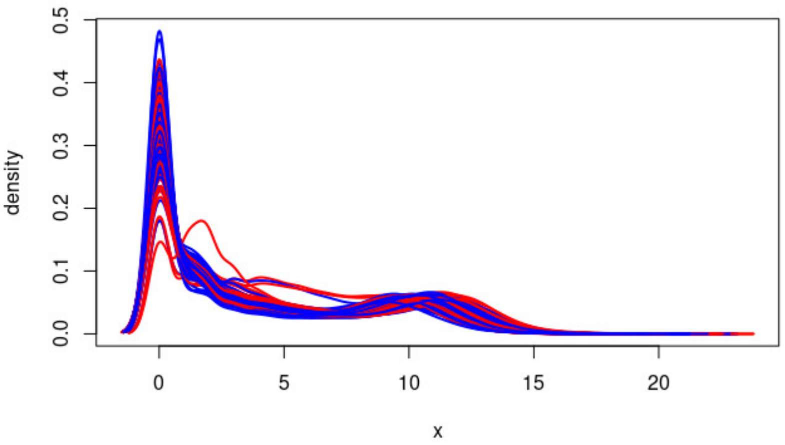
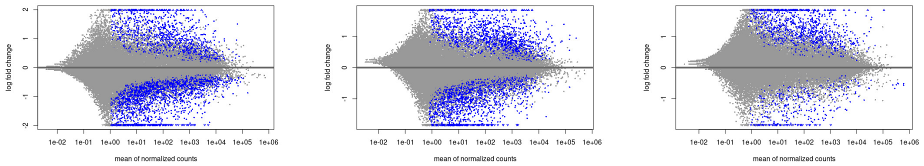


Figure 14: Density plot of survival and non-survival

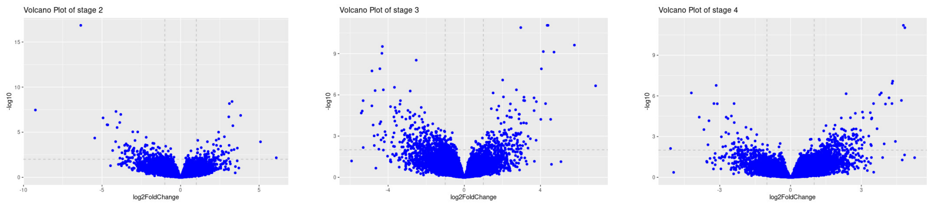


(a) Mean of normalized counts stage 2

(b) Mean of normalized counts stage 3

(c) Mean of normalized counts stage 4

Figure 15: Mean of normalized counts on the stages 2,3 and 4



(a) Volcano plot of stage 2 cancer

(b) Volcano plot of stage 3 cancer

(c) Volcano plot of stage 4 cancer

Figure 16: Volcano plot of stage 2, 3 and 4 cancer

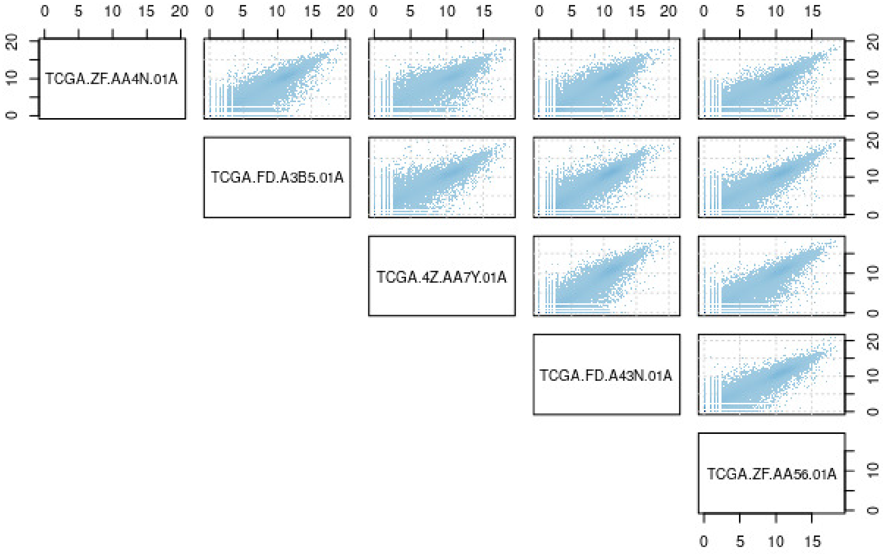


Figure 17: Correlation pairplot of log2 transformed counts

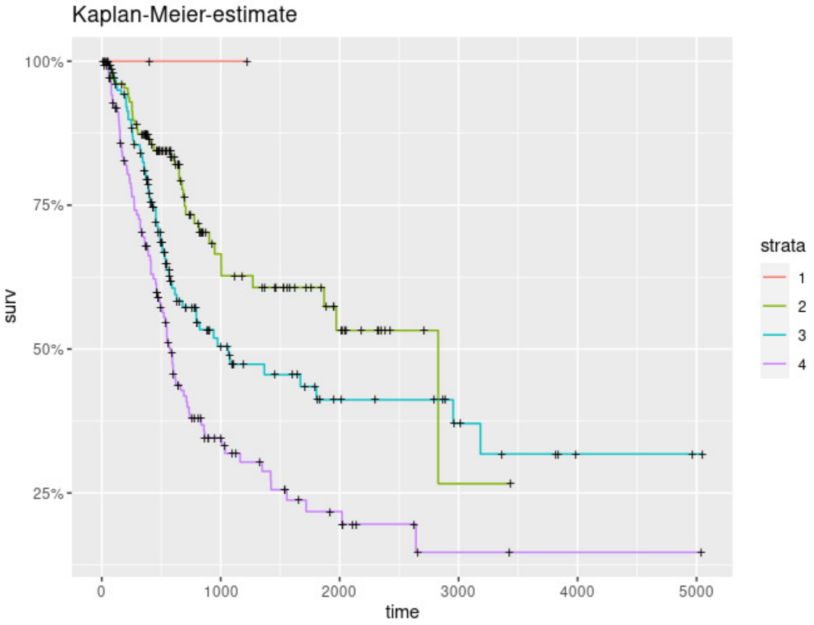


Figure 18: Kaplan Meier survival plot of all stages

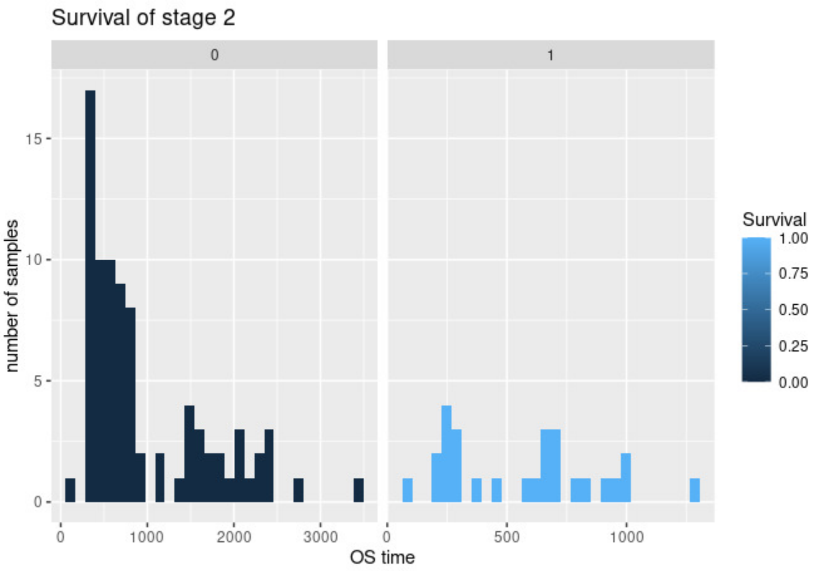


Figure 19: Survival OS time Barplot stage 2

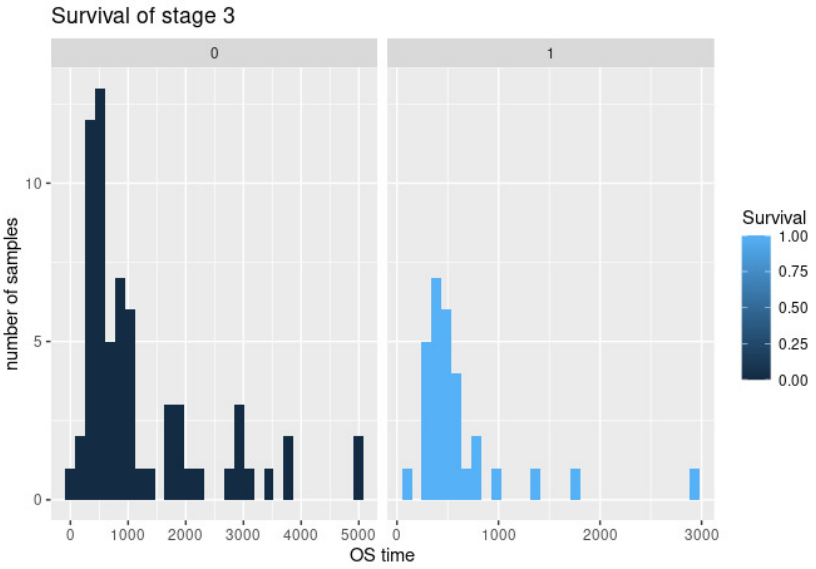


Figure 20: Survival OS time Barplot stage 3

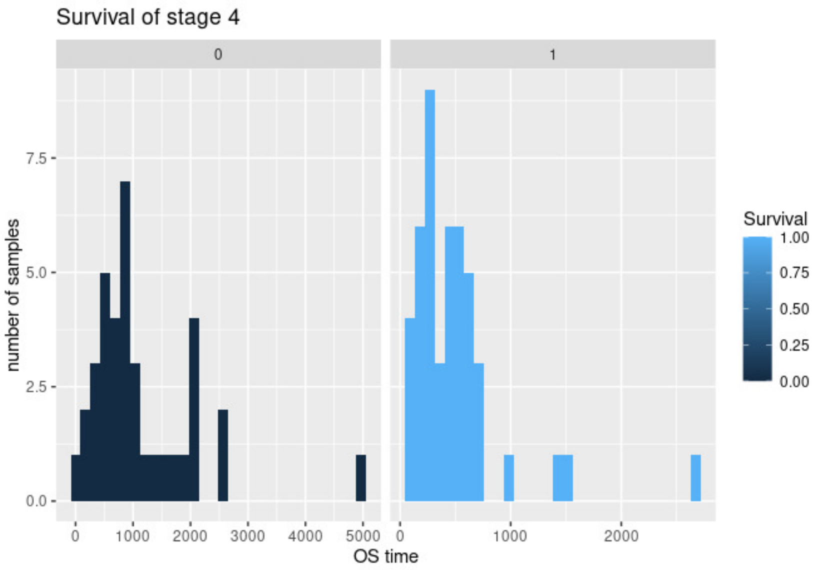


Figure 21: Survival OS time Barplot stage 4

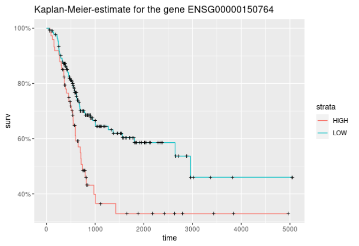


Figure 22: Kaplan Meier survival plot for the gene DIXDC1 in all stages

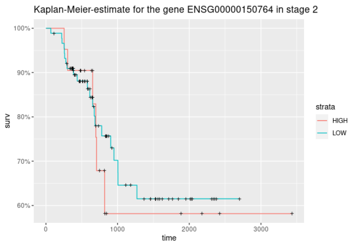


Figure 23: Kaplan Meier survival plot for the gene DIXDC1 in stage 2

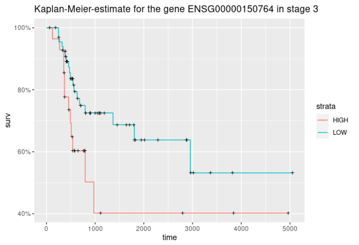


Figure 24: Kaplan Meier survival plot for the gene DIXDC1 in stage 3

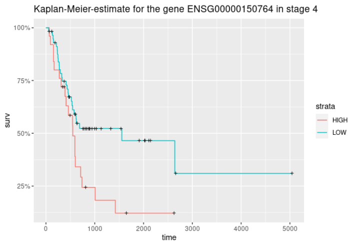


Figure 25: Kaplan Meier survival plot for the gene DIXDC1 in stage 4

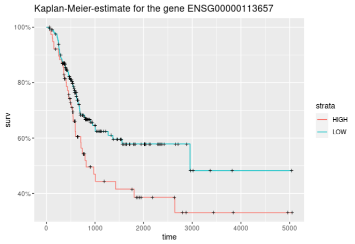


Figure 26: Kaplan Meier survival plot for the gene DPYSL3 in all stages

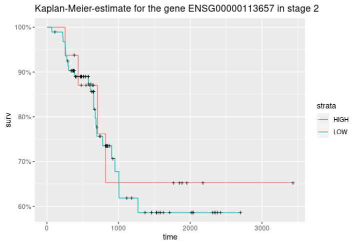


Figure 27: Kaplan Meier survival plot for the gene DPYSL3 in stage 2

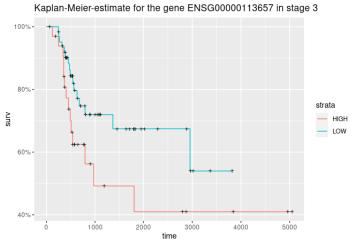


Figure 28: Kaplan Meier survival plot for the gene DPYSL3 in stage 3

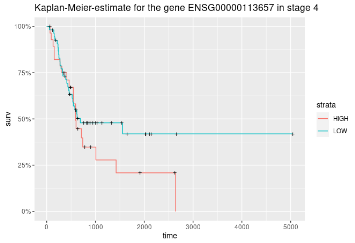


Figure 29: Kaplan Meier survival plot for the gene DPYSL3 in stage 4

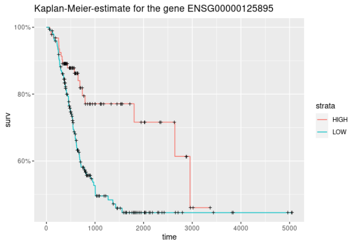


Figure 30: Kaplan Meier survival plot for the gene TMEM74 in all stages

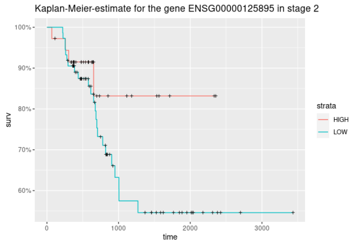


Figure 31: Kaplan Meier survival plot for the gene TMEM74 in stage 2

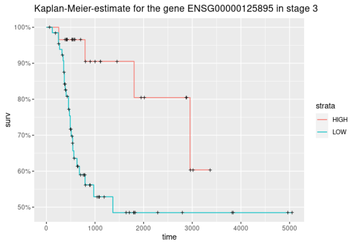


Figure 32: Kaplan Meier survival plot for the gene TMEM74 in stage 3

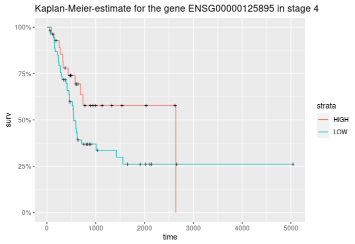


Figure 33: Kaplan Meier survival plot for the gene TMEM74 in stage 4

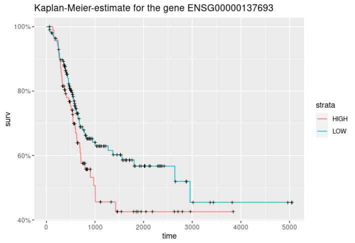


Figure 34: Kaplan Meier survival plot for the gene YAP1 in all stages

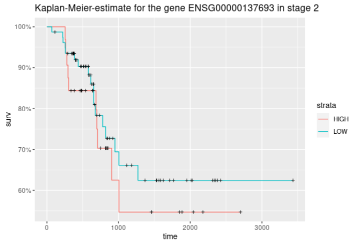


Figure 35: Kaplan Meier survival plot for the gene YAP1 in stage 2

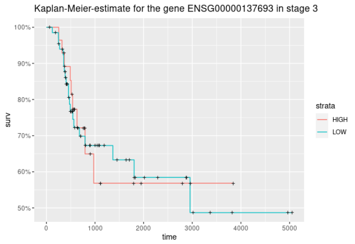


Figure 36: Kaplan Meier survival plot for the gene YAP1 in stage 3

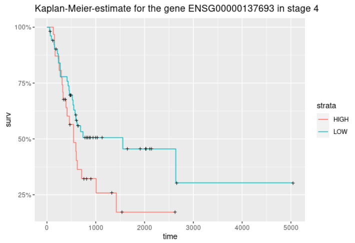


Figure 37: Kaplan Meier survival plot for the gene YAP1 in stage 4

Author details

References

1. Steffen Durinck, Paul Spellman, Ewan Birney, and Wolfgang Huber. Mapping identifiers for the integration of genomic datasets with the r/bioconductor package biomart. *Nature Protocols*, 4:1184–1191, 2009.
2. Jerome Friedman, Robert Tibshirani, and Trevor Hastie. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22, 2010.
3. Michael Lawrence, Wolfgang Huber, Hervé Pagès, Patrick Aboyoun, Marc Carlson, Robert Gentleman, Martin Morgan, and Vincent Carey. Software for computing and annotating genomic ranges. *PLoS Computational Biology*, 9, 2013.
4. PI Liang, HY Lai, TC Chan, et al. Upregulation of dihydropyrimidinase-like 3 (dpysl3) protein predicts poor prognosis in urothelial carcinoma. *BMC Cancer*, 23:599, 2023. Published on July 23, 2023.
5. Sandro Morganella, Stefano Maria Pagnotta, and Michele Ceccarelli. Finding recurrent copy number alterations preserving within-sample homogeneity. *Bioinformatics*, 27(21):2949–2956, 2011. Published: 25 August 2011.
6. National Cancer Institute. The cancer genome atlas (tcga) code tables, Accessed: January 17, 2024.
7. UH Weidle and F Birzele. Bladder cancer-related micrornas with in vivo efficacy in preclinical models. *Cancer Diagn Progn*, 1(4):245–263, 2021. Published on July 3, 2021.
8. Y Wu, Q Zheng, Y Li, et al. Metformin targets a yap1-tead4 complex via ampk to regulate ccne1/2 in bladder cancer cells. *Journal of Experimental & Clinical Cancer Research*, 38:376, 2019.
9. Z Xu, H Chen, J Sun, W Mao, S Chen, and M Chen. Multi-omics analysis identifies a lncrna-related prognostic signature to predict bladder cancer recurrence. *Bioengineered*, 12(2):11108–11125, 2021.
10. C Zhou, AH Li, S Liu, and H Sun. Identification of an 11-autophagy-related-gene signature as promising prognostic biomarker for bladder cancer patients. *Biology (Basel)*, 10(5):375, 2021. Published on April 27, 2021.