[Predictive Model for Early Cancer Detection]

[G11]

**Data Science Capstone Project   
Predictive Modeling Report**

Date:

[03/12/2021]

Team Members:

Name: Ryner Menezes (RPM87)

Name: Astha Jain (AJ887)

Name: Sudharshan Rao (SB4267)

Name: Divya Reddy (DL957)

Name: Madhu Bandru (MB4236)

# Contents

[Contents 2](#_Toc66435770)

[1. DEFINE THE PREDICTIVE MODELING PROBLEM 4](#_Toc66435771)

[1.1 Input: What are the input data and define the input data clearly? 4](#_Toc66435772)

[1.1.1 Inputs for Model 1 (Binary Classification) 4](#_Toc66435773)

[1.1.2 Inputs for Second Model (Multi-Classification) 4](#_Toc66435774)

[1.2. Data Representation: What is the data representation? 5](#_Toc66435775)

[1.2.1 Justification for Inputs Selected 9](#_Toc66435776)

[1.3 Output: What are you trying to predict? Define the output clearly. 17](#_Toc66435777)

[2. PREDICTIVE MODELS 18](#_Toc66435778)

[2.1 What are the methods? Give a general introduction of the methods with references. 18](#_Toc66435779)

[2.1.2 Training Process for Binary Classification 18](#_Toc66435780)

[2.1.3 Training Process for Multi-Classification 20](#_Toc66435781)

[2.1.4 Over-Sampling 22](#_Toc66435782)

[2.2 Describe the methods with a pseudo code using the definitions in Section 1. 22](#_Toc66435783)

[2.2.1 Pseudo code - Random Forest Classifier model: 22](#_Toc66435784)

[2.2.2 Pseudo code – Deep Learning – Feed Forward Neural Network: 24](#_Toc66435785)

[2.3 Justify the Choice of the Method. 26](#_Toc66435786)

[2.3.1 Goal 26](#_Toc66435787)

[2.3.2 Training Process 26](#_Toc66435788)

[2.3.3 Choosing the right Model for our Data 27](#_Toc66435789)

[3.MODEL PERFORMANCE AND EVALUATIONS 28](#_Toc66435790)

[3.1 What metrics do you use for evaluation? 28](#_Toc66435791)

[3.2 What is your ground truth? 28](#_Toc66435792)

[3.3 Discuss the performance and the limitation of the method. 28](#_Toc66435793)

[3.3.1 Performance for Binary Classification 28](#_Toc66435794)

[3.3.2 Performance for Multi Classification 29](#_Toc66435795)

[3.4 Diagnostics and Limitations 33](#_Toc66435796)

[4.CONCLUSION 34](#_Toc66435797)

[5.APPENDIX 34](#_Toc66435798)

[4.1 Feed forward Neural network binary Classifier using Keras 35](#_Toc66435799)

[4.2 Feed forward Neural network binary Classifier using Keras 35](#_Toc66435800)

[5. REFERENCES 35](#_Toc66435801)

[Table of Contributions 36](#_Toc66435802)

[The purpose of this report is to describe the predictive modeling on the data that you have acquired, pre-processed, and explored in DSCI 591.]

# 1. DEFINE THE PREDICTIVE MODELING PROBLEM

1. Input: What are the input data and define the input data clearly?
2. Data Representation: What is the data representation?
3. Output: What are you trying to predict? Define the output clearly.

## 1.1 Input: What are the input data and define the input data clearly?

There are 2 types of predictions to be made –

1. Model 1 – Predicting if a person is cancer positive or negative (binary classification)
2. Model 2 – Localization of cancer (multi-classification).

The input data to be fed into the model will be different for each type of classification.

## 1.1.1 Inputs for Model 1 (Binary Classification)

First model takes 9 inputs, which includes Omega score - measurement of the levels of the Omega-3 Fatty Acids in a blood sample and eight circulating protein biomarkers. These protein biomarkers were based on the key features identified from boxplots during exploratory data analysis and from domain literature [1].

|  |  |
| --- | --- |
| **Input Field** | **Input Field Type** |
| OmegaScore | Continuous |
| CA-125 (U/ml) | Continuous |
| CEA (pg/ml) | Continuous |
| CA19-9 (U/ml) | Continuous |
| Prolactin (pg/ml) | Continuous |
| HGF (pg/ml) | Continuous |
| OPN (pg/ml) | Continuous |
| Myeloperoxidase (ng/ml) | Continuous |
| TIMP-1 (pg/ml) | Continuous |

**Table 1 -Inputs for Binary Classification Model**

### 1.1.2 Inputs for Second Model (Multi-Classification)

Second model takes 41 inputs which includes, Sex, Omega Score and 39 circulating protein biomarkers.

|  |  |
| --- | --- |
| **Variable Name** | **Variable Type** |
| Sex | Nominal |
| OmegaScore | Continuous |
| AFP (pg/ml) | Continuous |
| Angiopoietin-2 (pg/ml) | Continuous |
| AXL (pg/ml) | Continuous |
| CA-125 (U/ml) | Continuous |
| CA 15-3 (U/ml) | Continuous |
| CA19-9 (U/ml) | Continuous |
| CD44 (ng/ml) | Continuous |
| CEA (pg/ml) | Continuous |
| CYFRA 21-1 (pg/ml) | Continuous |
| DKK1 (ng/ml) | Continuous |
| Endoglin (pg/ml) | Continuous |
| FGF2 (pg/ml) | Continuous |
| Follistatin (pg/ml) | Continuous |
| Galectin-3 (ng/ml) | Continuous |
| G-CSF (pg/ml) | Continuous |
| GDF15 (ng/ml) | Continuous |
| HE4 (pg/ml) | Continuous |
| HGF (pg/ml) | Continuous |
| IL-6 (pg/ml) | Continuous |
| IL-8 (pg/ml) | Continuous |
| Kallikrein-6 (pg/ml) | Continuous |
| Leptin (pg/ml) | Continuous |
| Mesothelin (ng/ml) | Continuous |
| Midkine (pg/ml) | Continuous |
| Myeloperoxidase (ng/ml) | Continuous |
| NSE (ng/ml) | Continuous |
| OPG (ng/ml) | Continuous |
| OPN (pg/ml) | Continuous |
| PAR (pg/ml) | Continuous |
| Prolactin (pg/ml) | Continuous |
| sEGFR (pg/ml) | Continuous |
| sFas (pg/ml) | Continuous |
| SHBG (nM) | Continuous |
| sHER2/sEGFR2/sErbB2 (pg/ml) | Continuous |
| sPECAM-1 (pg/ml) | Continuous |
| TGFa (pg/ml) | Continuous |
| Thrombospondin-2 (pg/ml) | Continuous |
| TIMP-1 (pg/ml) | Continuous |
| TIMP-2 (pg/ml) | Continuous |

**Table 2 -Inputs for Multi Classification Model to Predict Cancer Type**

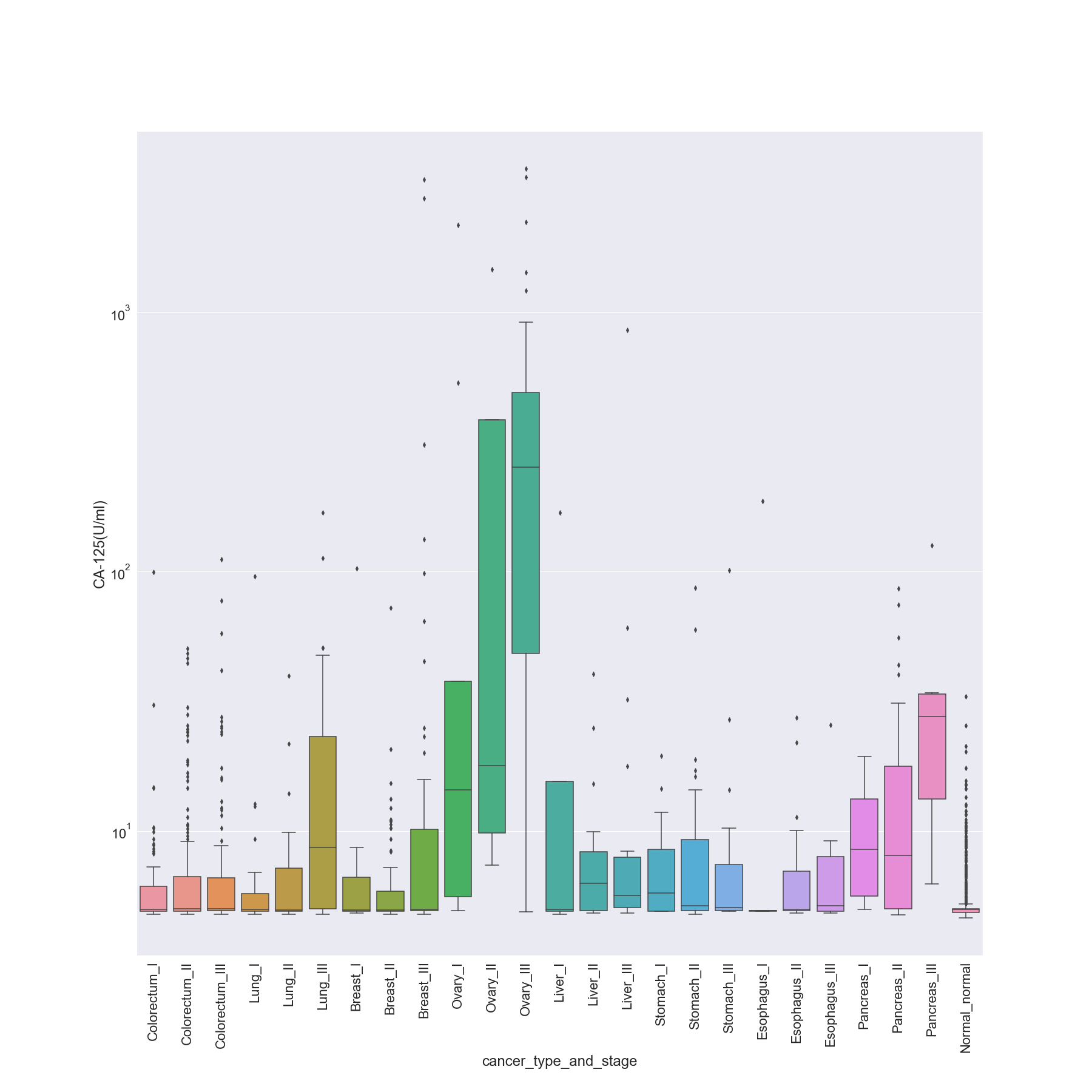
## 1.2. Data Representation: What is the data representation?

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Variable Name** | **Variable Type** | **count** | **mean** | **std** | **min** | **25%** | **50%** | **75%** | **max** |
| **Age** | Numeric Discrete | 1817 | 56.80793 | 17.31545 | 17 | 47 | 60 | 69 | 93 |
| **Plasma volume (mL)** | Numeric and Continuous | 1817 | 7.369356 | 0.623067 | 2 | 7.5 | 7.5 | 7.5 | 7.5 |
| **Plasma DNA concentration (ng/mL)** | Numeric and Continuous | 1817 | 8.924948 | 15.17635 | 0 | 2.31 | 4.38 | 8.2 | 157.48 |
| **Î© score** | Numeric and Continuous | 1817 | 4.278426 | 20.79013 | 0 | 0.68 | 0.95 | 1.29 | 333.23 |
| **Mutant allele frequency (%)** | Numeric and Continuous | 1817 | 0.630958 | 3.956516 | 0 | 0.03 | 0.05 | 0.09 | 62.32 |
| **Mutant fragments/mL plasma** | Numeric and Continuous | 1817 | 12.51778 | 65.64388 | 0 | 0.3 | 0.7 | 1.6 | 862.8 |
| **AFP (pg/ml)** | Numeric and Continuous | 1817 | 7109.351 | 52353.92 | 706.158 | 829.98 | 946.938 | 1848.54 | 600608.9 |
| **Angiopoietin-2 (pg/ml)** | Numeric and Continuous | 1817 | 1908.423 | 1814.768 | 38.391 | 997.49 | 1498.92 | 2259.46 | 30001.79 |
| **AXL (pg/ml)** | Numeric and Continuous | 1811 | 2367.282 | 1369.703 | 109.44 | 1479.425 | 2136.61 | 2931.68 | 12247.31 |
| **CA-125 (U/ml)** | Numeric and Continuous | 1817 | 25.18304 | 184.5854 | 4.608 | 4.89 | 4.98 | 6.4 | 3600.024 |
| **CA 15-3 (U/ml)** | Numeric and Continuous | 1817 | 20.62861 | 64.34536 | 1.32 | 7.11 | 12.18 | 19.84 | 1177.446 |
| **CA19-9 (U/ml)** | Numeric and Continuous | 1817 | 53.82877 | 409.031 | 14.214 | 16.32 | 16.482 | 18.6 | 12491.47 |
| **CD44 (ng/ml)** | Numeric and Continuous | 1811 | 19.53303 | 11.34154 | 6.75 | 11.96 | 16.76 | 23.795 | 148.44 |
| **CEA (pg/ml)** | Numeric and Continuous | 1817 | 1857.966 | 16139.13 | 1 | 83 | 604.85 | 1062.12 | 337245.4 |
| **CYFRA 21-1 (pg/ml)** | Numeric and Continuous | 1817 | 4843.461 | 42382.34 | 1816.458 | 1955.244 | 1994.874 | 2106.97 | 1475728 |
| **DKK1 (ng/ml)** | Numeric and Continuous | 1817 | 1.050809 | 0.442416 | 0.35 | 0.74 | 0.94 | 1.25 | 5.97 |
| **Endoglin (pg/ml)** | Numeric and Continuous | 1817 | 1671.744 | 869.0901 | 79.05 | 1127.81 | 1600 | 2069.14 | 16244.26 |
| **FGF2 (pg/ml)** | Numeric and Continuous | 1817 | 141.7921 | 68.31556 | 80.274 | 91.062 | 116.7 | 174.27 | 734.55 |
| **Follistatin (pg/ml)** | Numeric and Continuous | 1817 | 887.8785 | 647.6738 | 62.22 | 487.91 | 765.99 | 1100.97 | 8126.49 |
| **Galectin-3 (ng/ml)** | Numeric and Continuous | 1817 | 7.945883 | 9.198508 | 0.2 | 4.3 | 5.86 | 8.31 | 140.43 |
| **G-CSF (pg/ml)** | Numeric and Continuous | 1810 | 195.012 | 413.9372 | 29.481 | 38.797 | 115.65 | 186.4225 | 12827.98 |
| **GDF15 (ng/ml)** | Numeric and Continuous | 1817 | 0.721519 | 1.031722 | 0.04 | 0.24 | 0.46 | 0.84 | 24.29 |
| **HE4 (pg/ml)** | Numeric and Continuous | 1817 | 5615.871 | 8440.005 | 3671.556 | 3997.188 | 4092.972 | 4209.156 | 189497.5 |
| **HGF (pg/ml)** | Numeric and Continuous | 1817 | 323.8637 | 487.681 | 158.334 | 164.514 | 183.58 | 293.15 | 11432.98 |
| **IL-6 (pg/ml)** | Numeric and Continuous | 1817 | 27.54403 | 94.27772 | 2.946 | 3.606 | 5.87 | 18.13 | 2818.46 |
| **IL-8 (pg/ml)** | Numeric and Continuous | 1817 | 31.06708 | 196.3903 | 7.56 | 8.178 | 8.83 | 19.85 | 5289.6 |
| **Kallikrein-6 (pg/ml)** | Numeric and Continuous | 1811 | 5258.886 | 3077.387 | 136.57 | 3484.195 | 4846.51 | 6402.565 | 53356.84 |
| **Leptin (pg/ml)** | Numeric and Continuous | 1817 | 28273.58 | 40550.64 | 727.182 | 4460.8 | 13172.35 | 35088.45 | 449756.6 |
| **Mesothelin (ng/ml)** | Numeric and Continuous | 1811 | 22.66759 | 21.63923 | 1.49 | 13.315 | 18.54 | 27.195 | 583.25 |
| **Midkine (pg/ml)** | Numeric and Continuous | 1811 | 603.754 | 1919.853 | 64.17 | 238.505 | 352.23 | 554.925 | 53954.89 |
| **Myeloperoxidase (ng/ml)** | Numeric and Continuous | 1817 | 31.19939 | 68.25568 | 1.3 | 8.05 | 12.83 | 22.63 | 1001 |
| **NSE (ng/ml)** | Numeric and Continuous | 1817 | 20.42283 | 22.80721 | 1.1 | 7.12 | 11.56 | 25.79 | 220.38 |
| **OPG (ng/ml)** | Numeric and Continuous | 1817 | 0.542944 | 0.606039 | 0.09 | 0.29 | 0.39 | 0.58 | 4.2 |
| **OPN (pg/ml)** | Numeric and Continuous | 1817 | 56295.36 | 48269.01 | 3218.166 | 26146.14 | 41236.83 | 68644.7 | 433959.6 |
| **PAR (pg/ml)** | Numeric and Continuous | 1811 | 7751.386 | 4998.79 | 663.27 | 4279.935 | 6649.42 | 10070.39 | 49041.88 |
| **Prolactin (pg/ml)** | Numeric and Continuous | 1817 | 32313.98 | 54139.46 | 806.28 | 8617.16 | 14032.92 | 26552.97 | 608432.4 |
| **sEGFR (pg/ml)** | Numeric and Continuous | 1811 | 2206.284 | 1214.316 | 197.58 | 1323.52 | 2052.17 | 2906.49 | 8576.92 |
| **sFas (pg/ml)** | Numeric and Continuous | 1816 | 1390.84 | 2354.801 | 192.948 | 206.334 | 1126.515 | 1803.705 | 61146.1 |
| **SHBG (nM)** | Numeric and Continuous | 1817 | 67.92226 | 54.47899 | 1.5 | 31.43 | 53.35 | 87.35 | 478.84 |
| **sHER2/sEGFR2/sErbB2 (pg/ml)** | Numeric and Continuous | 1811 | 5765.088 | 4376.222 | 306.28 | 4228.43 | 5261.22 | 6470.335 | 150848.1 |
| **sPECAM-1 (pg/ml)** | Numeric and Continuous | 1811 | 5883.916 | 2174.284 | 219.83 | 4384.605 | 5499.78 | 7023.805 | 20178.17 |
| **TGFa (pg/ml)** | Numeric and Continuous | 1817 | 28.10876 | 283.2641 | 15.258 | 16.2 | 16.488 | 16.698 | 12018.86 |
| **Thrombospondin-2 (pg/ml)** | Numeric and Continuous | 1811 | 5502.45 | 10204.12 | 482.14 | 1145.1 | 2245.65 | 5673.605 | 157461.1 |
| **TIMP-1 (pg/ml)** | Numeric and Continuous | 1817 | 70058.42 | 47577.49 | 976.55 | 41231.36 | 59282.78 | 82928.93 | 569512.7 |
| **TIMP-2 (pg/ml)** | Numeric and Continuous | 1817 | 40261.12 | 12970.48 | 15026.32 | 30752.35 | 37735.41 | 46794.54 | 105748.6 |

**Table 3 – Data Representation of all Input Features**

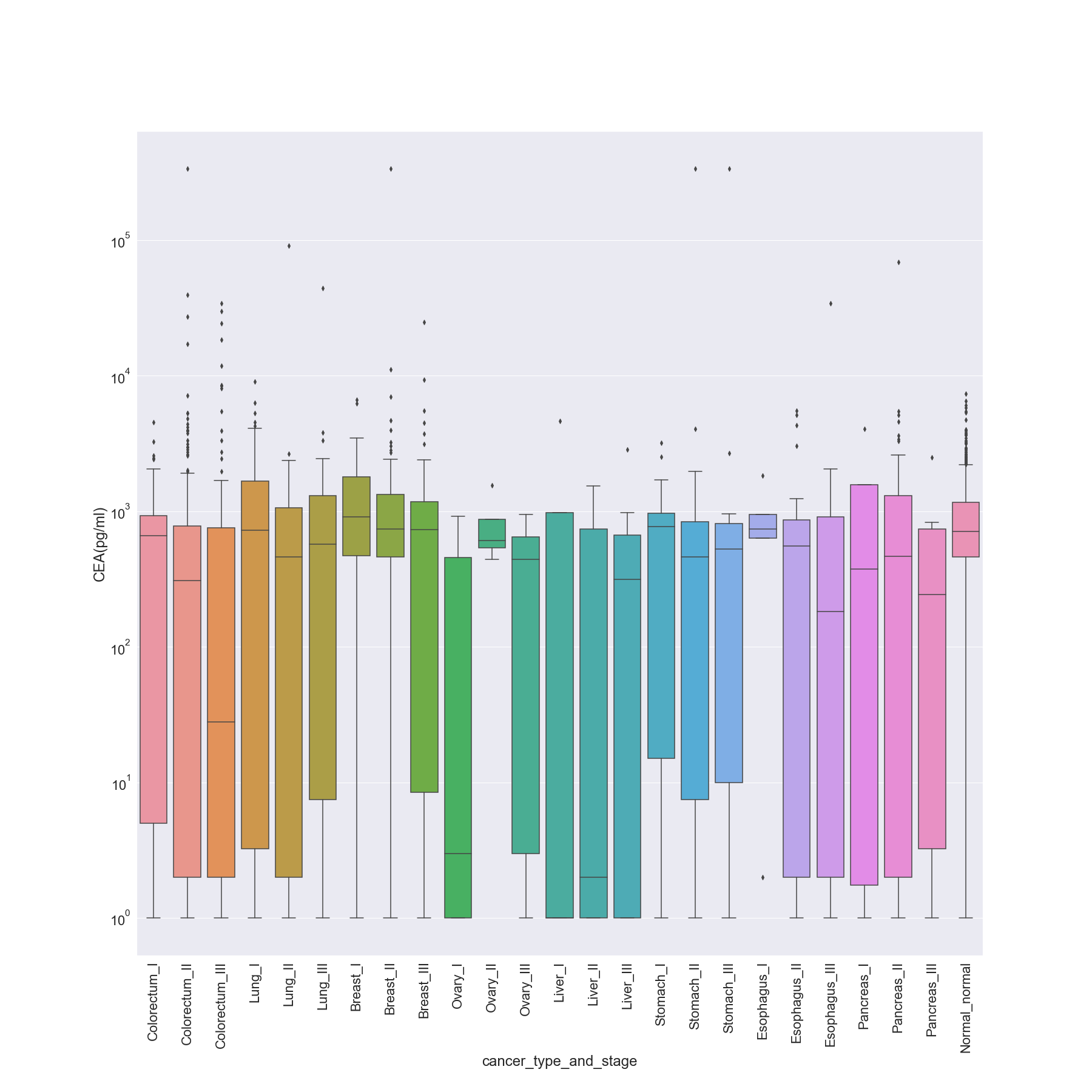
### 1.2.1 Justification for Inputs Selected

For the binary classification model, we have only chosen the 8 key protein biomarkers along with the ‘omega’ score instead of selecting all the protein biomarkers. The 7 key proteins were identified from the boxplots drawn at the exploratory data analysis stage and their distribution with respect to each cancer types and from the knowledge gained in domain literature [1][2][3]. They have been clearly justified as each of the protein biomarkers are useful in detecting different cancer types and combining them together will help us in identifying if a person has cancer or not. The lean set of features will also help to scale the model easily. For multiclassification though, we would need all the 41 protein biomarkers to localize the type of cancer for the model to perform well.



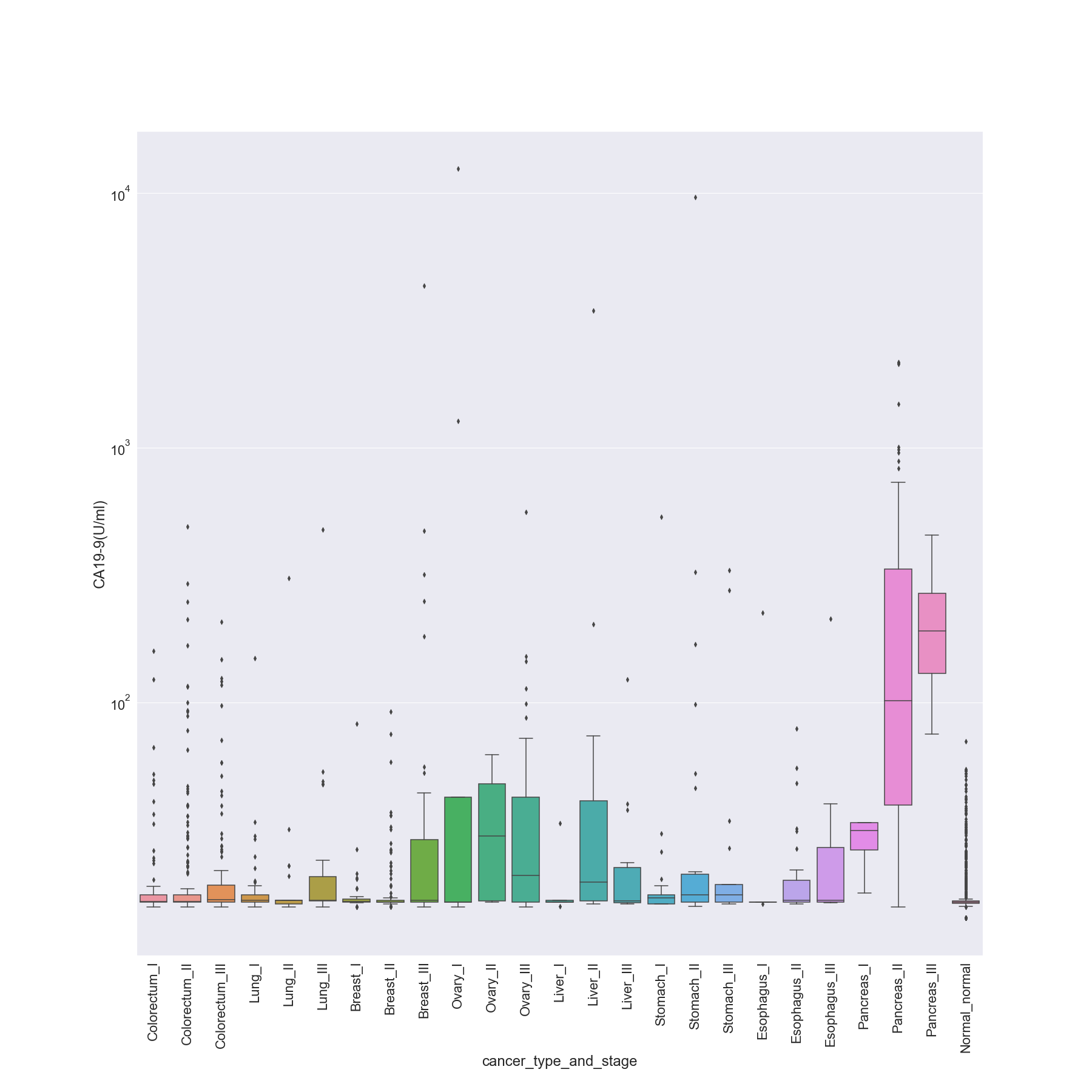
**Figure 1 – Distribution for Protein Bio marker CA-125(U/ml)**

From figure 1, we can see that CA-125 protein biomarker has a predominant presence in ovary cancer patients and is useful in identifying ovary cancer.



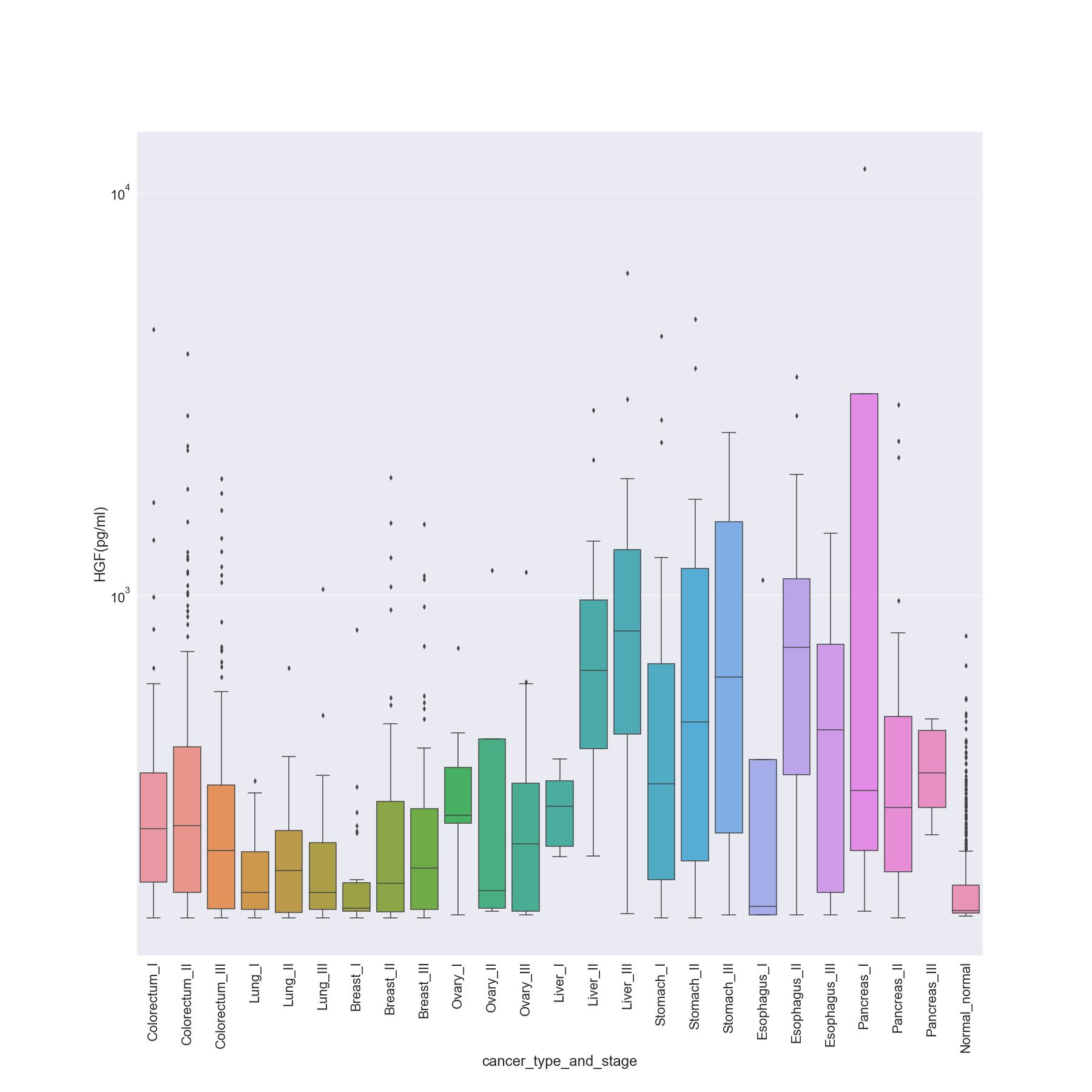
**Figure 2 – Distribution for Protein Bio marker CEA(pg/ml)**

From figure 2, we can see CEA is a protein biomarker that is responsible for most types of cancer since it is found in wide concentration levels especially in lower levels, in many types of cancers like stomach, esophagus, and pancreas, compared to normal patients who do not have lower concentration levels of this protein.



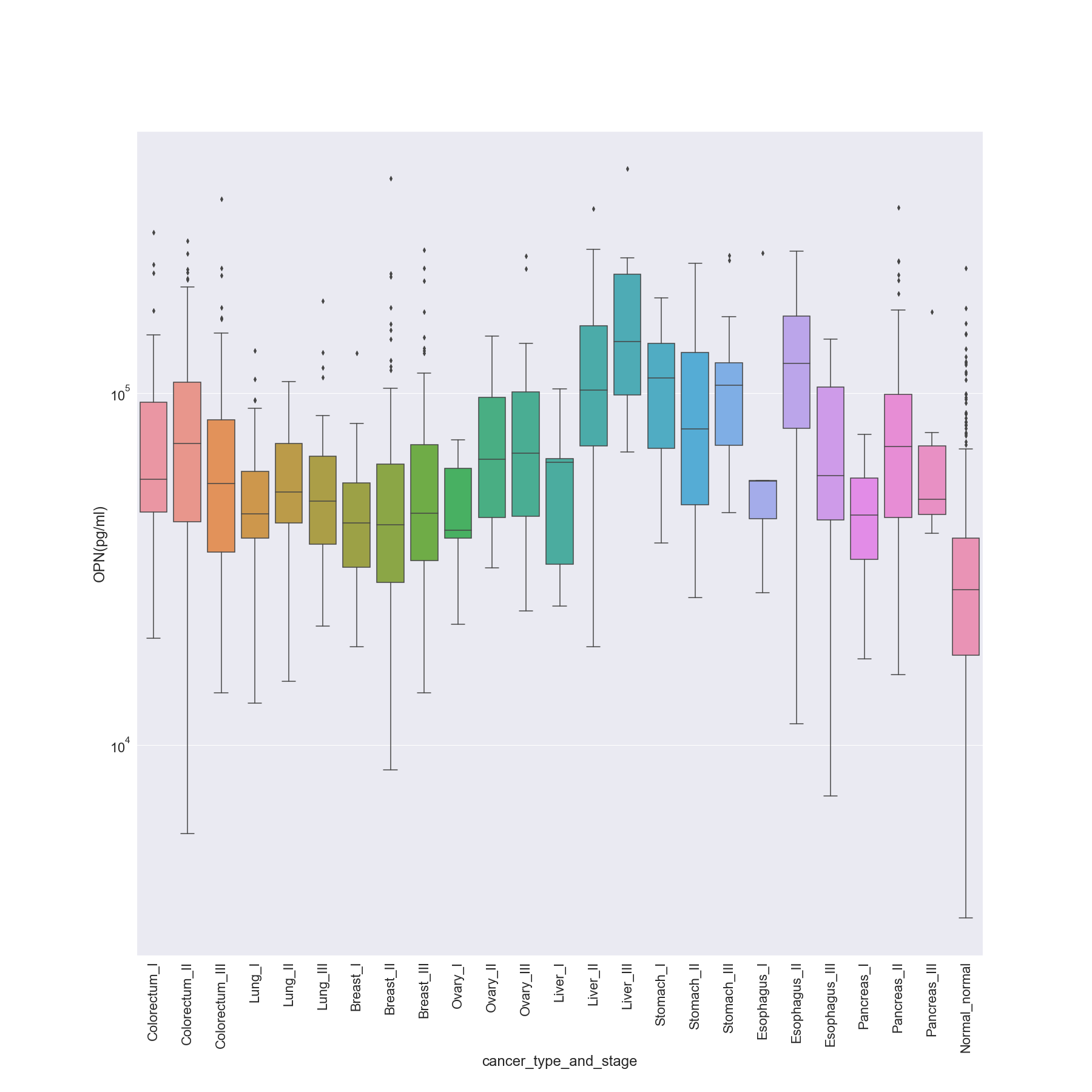
**Figure 3 – Distribution for Protein Bio marker CA19-9(U/ml)**

From figure 3, we can see that CA19-9 is useful in detecting pancreas cancer, since the concentration of this protein is higher in the patients detected with pancreatic cancer.



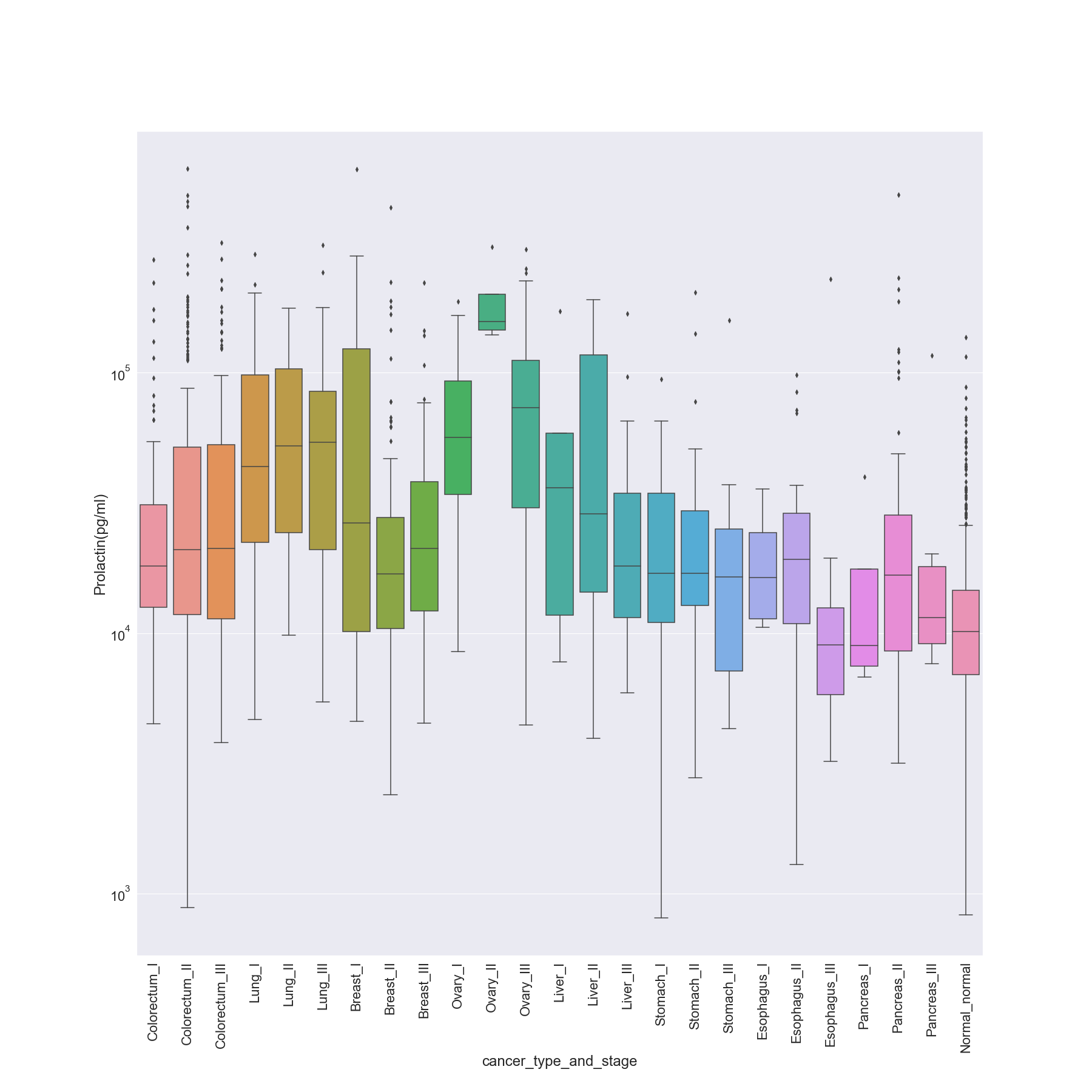
**Figure 4 – Distribution for Protein Bio marker HGF (pg/ml)**

From figure 4, we can see that HGF concentration is higher in Patients of Stomach, Esophagus, and pancreatic cancer.



**Figure 5 – Distribution for Protein Bio marker OPN (pg/ml)**

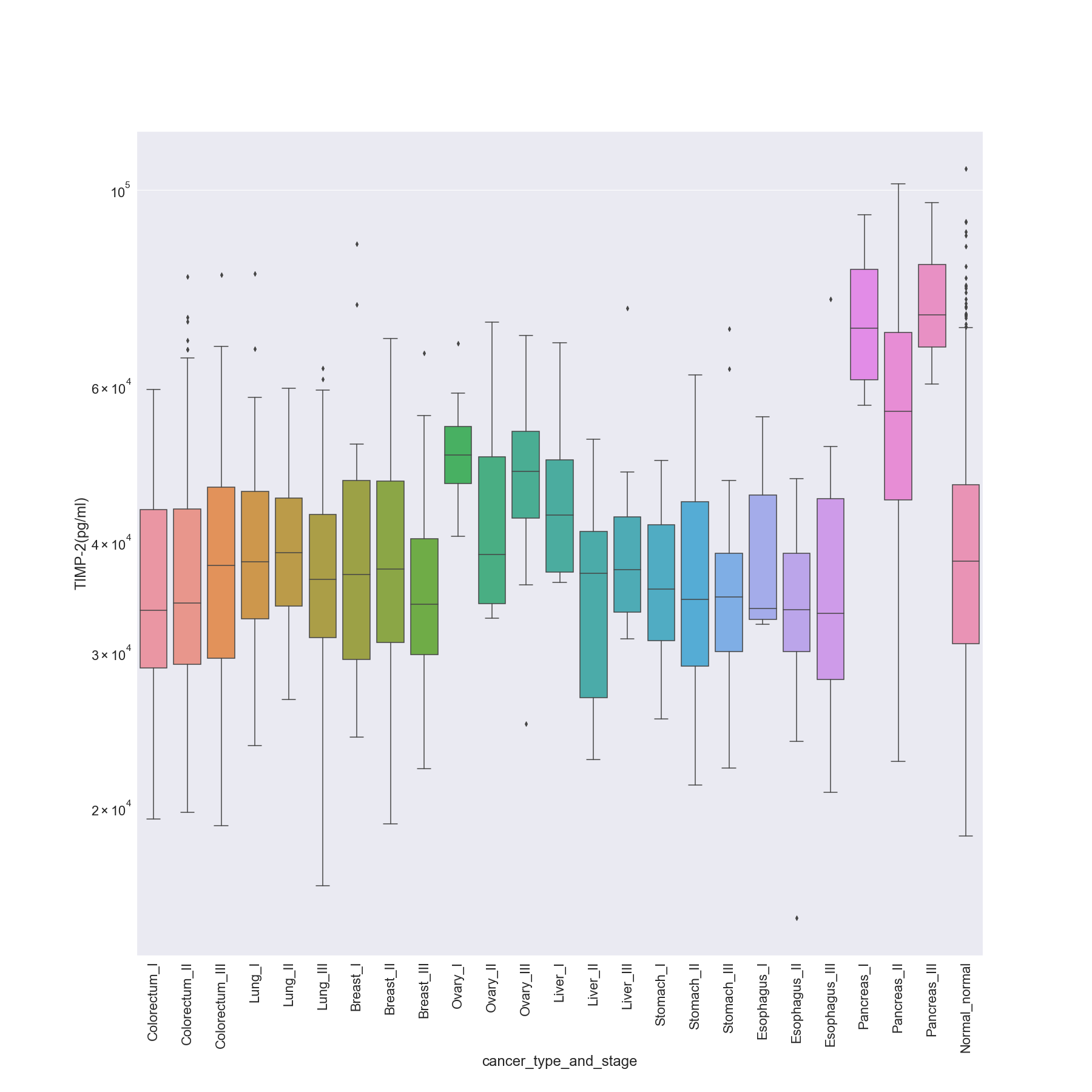
From figure 5, we can see that OPN is useful in detecting Liver and Esophagus cancer.



**Figure 6– Distribution for Protein Bio marker Prolactin (pg/ml)**



**Figure 7– Distribution for Protein Bio marker Myeloperoxidase (ng/ml)**



**Figure 8– Distribution for Protein Bio marker TIMP-2 (pg/ml)**

From figure 8, we see that TIMP-2 is useful for detecting Pancreas cancer.

These 9 key protein biomarkers stand out while compared to other protein biomarkers in their distribution with respect to different cancer types. This makes them easier to associate with a specific cancer because of their distinct distribution.

## Output: What are you trying to predict? Define the output clearly.

As mentioned in section 1.1, we intend to build two models and both the models have one output which is a prediction based on the following -

1. **Model 1- The Binary classification model** can be used to detect if the cancer is present given the concentration of the 7 key protein biomarkers. The output for this model is a binary class which can take either ‘positive’ or ‘negative’ values. Positive indicates that cancer is present on the test instance and negative means there is no cancer. Our dataset has a labelled column name ‘Target’ which will be used for the ground truth labels while training the algorithm.
2. **Model 2 – The multiclassification model** with 8 classes for localization of cancer, which means the type of cancer is detected, out of eight common cancers - ovary, liver, esophagus, pancreas, stomach, colorectal, lung, and breast cancer. The output will take only a single value from the above 8 classes. Our dataset has a labelled column ‘Tumor\_Type’ which has the ground truth labels and will be used for training the multiclassification mode.

In case if the first model detects no cancer, then the second model is not invoked. The dataset does not provide any information on survival from cancer for the patients. The goal is to build a strong predictive model that can be helpful to identify cancer in early stages so that it can be treated on time.

# 2. PREDICTIVE MODELS

## 2.1 What are the methods? Give a general introduction of the methods with references.

The methods we have chosen from machine learning classification algorithms are **random forest and deep learning.** Our initial plan was to use these for both binary and multi classification and the model with a higher baseline score was used to fine tune after week 5.

The justification for these methods chosen are explained in section 2.3.

* Random Forest – This algorithm is an estimator that fits number of decision tree classifiers on multiple samples of dataset. This helps to improve predictive accuracy by generalizing using the given training data without causing high variance. For the baseline model, we have used the default parameter value of n\_estimators = 100 which represents number of trees in the forest. The random forest is built using the scikit learn library.
* Feed Forward Neural Network - We are using keras to build our neural network. We import the keras library to create the neural network layers. There are two main types of models available in keras — Sequential and Dense library. We will use Sequential model to build our neural network. We use Dense library to build input, hidden and output layers of a neural network.
* We have also computed mutual information between features of training data which measures the information gain and helps us identify the most important features.

### 2.1.2 Training Process for Binary Classification

#### 2.1.2.1 Training Features with Information gain

We have used the method ‘Mutual Information’ which calculates the information gain each feature contributes to predicting the target variable. Higher the value, more important the feature for the target prediction. In the table 4 below, protein biomarkers CA19-9(U/ml), HGF, CA-125 are having higher scores and prolactin, Myeloperoxidase, TIMP-1 are having lower values, so they are almost independent on the target variable. With these values, we will be able to know about the variables that are correlated with the target variable.

|  |  |
| --- | --- |
| **Features** | **Mutual Information** |
| CA19-9(U/ml) | 0.49671 |
| HGF(pg/ml) | 0.39744 |
| CA-125(U/ml) | 0.39269 |
| OPN(pg/ml) | 0.2013 |
| CEA(pg/ml) | 0.19442 |
| Ω score | 0.14 |
| Prolactin(pg/ml) | 0.13079 |
| Myeloperoxidase(ng/ml) | 0.10931 |
| TIMP-1(pg/ml) | 0.09573 |

**Table 4 – Mutual Information of Input features with respect to Target variable for Binary Classification**

#### 2.1.2.2 Training and test split

For binary classification model, we have allocated 25 percent of data to the test set and 75 percent of data to the training set. We have also given random\_state value as 42 to maintain the same set of instances whenever the data is sliced. Once the data is sliced, we use the trained data to fit the models and predict the target variable with test data.

#### 2.1.2.3 Target distribution

The Target is binary – “Positive” and “Negative”. Positive indicates presence of cancer and negative indicates no cancer.

‘Positive’ values count: 1005 (69.07%)

‘Negative’ values count: 812 (44.69%)

Chart, histogram

Description automatically generated

**Figure 9– Distribution of Target Variable for Binary Classification**

### 2.1.3 Training Process for Multi-Classification

#### 2.1.3.1 Training features with Information Gain

The mutual information score for each of the feature considered is sorted in decreasing order and is represented in the table below.

|  |  |
| --- | --- |
| **Feature Name** | **Mutual Information Score** |
| TGFa(pg/ml) | 0.677883 |
| HE4(pg/ml) | 0.569816 |
| AFP(pg/ml) | 0.490902 |
| G-CSF(pg/ml) | 0.483873 |
| CYFRA-21-1(pg/ml) | 0.459095 |
| Thrombospondin-2(pg/ml) | 0.447904 |
| IL-6(pg/ml) | 0.441493 |
| CA19-9(U/ml) | 0.431875 |
| sFas(pg/ml) | 0.382452 |
| CA-125(U/ml) | 0.306149 |
| FGF2(pg/ml) | 0.274153 |
| IL-8(pg/ml) | 0.260886 |
| HGF(pg/ml) | 0.197544 |
| Sex | 0.193392 |
| sHER2/sEGFR2/sErbB2(pg/ml) | 0.165629 |
| Ω score | 0.14745 |
| CD44(ng/ml) | 0.139161 |
| CA-15-3(U/ml) | 0.133966 |
| OPG(ng/ml) | 0.132451 |
| GDF15(ng/ml) | 0.122184 |
| Galectin-3(ng/ml) | 0.121462 |
| NSE(ng/ml) | 0.119014 |
| TIMP-2(pg/ml) | 0.118494 |
| TIMP-1(pg/ml) | 0.108689 |
| sEGFR(pg/ml) | 0.094286 |
| CEA(pg/ml) | 0.089708 |
| Midkine(pg/ml) | 0.082434 |
| Myeloperoxidase(ng/ml) | 0.079738 |
| Kallikrein-6(pg/ml) | 0.076084 |
| PAR(pg/ml) | 0.072096 |
| sPECAM-1(pg/ml) | 0.070427 |
| OPN(pg/ml) | 0.0698 |
| Leptin(pg/ml) | 0.068284 |
| Age | 0.066001 |
| AXL(pg/ml) | 0.06415 |
| Endoglin(pg/ml) | 0.057143 |
| Mesothelin(ng/ml) | 0.055191 |
| Prolactin(pg/ml) | 0.050563 |
| SHBG(nM) | 0.038056 |
| Follistatin(pg/ml) | 0.037941 |
| DKK1(ng/ml) | 0.030412 |
| Angiopoietin-2(pg/ml) | 0.020606 |

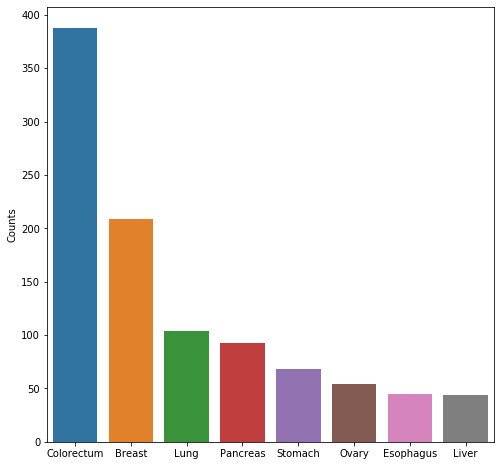
**Table 5 – Mutual Information of Input features with respect to Tumor type.**

#### 2.1.3.2 Training and test split

For the multi classification problem, we have dropped all rows of ‘normal’ or healthy individuals. This is because the predictive model to detect whether an individual has cancer or not is covered by the binary classification problem and we intend to invoke this model only on individuals who have been detected with cancer to localize the type of cancer.

The reduced dataset after removing the normal individuals has 1005 rows. We have split the data into 75-25, where 75% of the instances are included in training data and 25% instances belong to test data. We do not use stratified sampling here since for some type of cancers the data points are less in number and hence, we let the sampling be random.

#### 2.1.3.3 Target Distribution



**Figure 9– Distribution of Target ‘Tumor\_Type’ for multiclassification.**

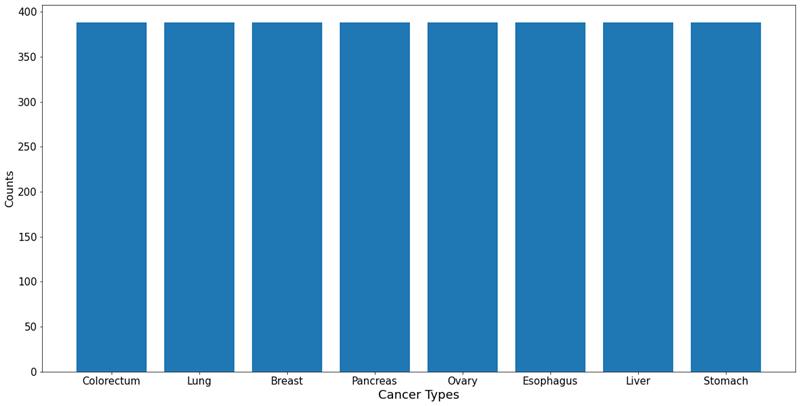
The number of classes across each type of cancer is not the same as the dataset is more biased towards colorectum and breast cancer.

### 2.1.4 Over-Sampling

As seen in figure 9, the distribution of target classes for the multiclassification problem is imbalanced. For this reason, we plan to oversample the data to balance these classes. However, we have also created a random forest model without sampling to compare performances between both.

The oversampling was done using ‘RandomOverSampler’ from sklearn and the skewed samples were oversampled on the threshold of maximum instances available which is of ‘colorectum’ cancer (388 instances, 38.6% of total instances). Figure 10 shows the distribution of all classes post over sampling.

#### 2.1.4.1 Class Distribution after Oversampling



**Figure 10– Distribution of Target ‘Tumor\_Type’ for multiclassification after Oversampling Data.**

## 2.2 Describe the methods with a pseudo code using the definitions in Section 1.

### 2.2.1 Pseudo code - Random Forest Classifier model:

Pseudo code for both binary and multiclass random forest is same, but the inputs and outputs differ.

#### 2.2.1.1 TRAINING:

**Input for Binary Classifier - Random Forest:**

Original data is split in 75% and 25% respectively for training and testing.

Shape of Train data: (1362, 9), (1362,1)

Where 1362 instances with 9 features and 1 target feature are feed to Random Forest Binary Classifier training.

**Input for Multi Classifier Random Forest:**

Shape of Train data: (753, 41), (753, 1)

Where 753 instances with 41 features and 1 target feature are feed to Random Forest Binary Classifier training.

**Algorithm:**

GENERATE BOOTSTRAP DATASET USING RANDOM SAMPLES FROM THE ORIGINAL DATASET

RANDOMLY SELECTS “K” FEATURES FROM TOTAL “M” FEATURES OF BOOTSTRAP DATASET, WHERE K<M

CALCULATE GINI INDEX OF “K” FEATURES

IF GINI INDEX OF A FEATURE IS LESS THAN K-1 FEATURES

CONSIDER IT AS ROOT NODE “D”

ELSE

MAKE THE FEATURE WITH LESS GINI INDEX AS ROOT NODE

ONCE ROOT NODE “D” IS FIXED, AND DATA SPLIT INTO INTERNAL NODES

CALCULATE GINI INDEX OF EACH FEATURE AT EACH INTERNAL NODE

IF GINI INDEX IS LOWER

THEN SPLIT DATA AT THAT NODE

ELSE

SPLIT DATA AT NODE WITH LESS GINI INDEX

.

.

.

REPEAT SAME STEPS UNTIL INTERNAL NODES ARE REACHED TO LEAF NODES.

.

.

.

REPEAT SAME STEPS TO BUILD FOREST FOR 100 (**N\_ESTIMATORS**) TREES.

Output from the above algorithm will be a trained model.

#### 2.2.1.2 PREDICTION:

**Input for Binary Classifier - Random Forest prediction:**

Shape of Test data: (455, 9)

Where 455 instances with 9 features are feed to Random Forest Binary Classifier predictor.

**Input for Multi Classifier Random Forest prediction:**

Shape of Test data: (252, 41)

Where 252 instances with 41 features are feed to Random Forest Multi Classifier predictor.

**Algorithm:**

ONCE 100 TREES ARE GENERATED

USE THE RULE OF EACH DECISION TREE TO PREDICT OUTCOME OF TEST SAMPLE AND STORE THE OUTCOME

CALCULATE VOTES FOR EACH PREDICTED TARGET

IF HIGH VOTES

FINAL PREDICTION

ELSE

IGNORE

**Output for Binary classifier – Random Forest:**

An array of classes (‘positive’, ‘negative’) for 455 instances.

**Output for Multi classifier – Random Forest:**

An array of classes (‘ovary’, ‘liver’, ‘esophagus’, ‘pancreas’, ‘stomach’, ‘colorectal’, ‘lung’, and ‘breast’) for 252 instances.

Steps for both binary and multiclass random forest is same. For multiclass random forest before modeling, over sampling is performed and same step (STEP 3A) mentioned below. Target variable for binary and multiclass random forest model are different.

**STEP1:** Import all necessary python packages

**STEP2:** Load preprocessed data set

**STEP3:** Separate input and target variable into two data frames

**STEP 3A (MULTICLASS ONLY):** Apply over sampling technique

For adjusting the class (8 tumor types) distribution over sampling is performed on the original dataset.

RandomOverSampler(random\_state=42)

**random\_state** controls randomization of the algorithm and 42 generate same sampling across multiple runs.

**STEP4:** Split train and test data using “train\_test\_split” function. 25% of the data is test data.

**STEP5:** load Random Forest Classifier with parameters **n\_estimators** = 100 and **n\_jobs** = -1

**n\_estimators** imply how many trees to be built in the forest. Higher number of trees gives better performance but makes code slower. Considering data size and after multiple runs, we selected its value as **100**

**n\_jobs** tell how many processors to be used. **“1”** means it can only use one processor where **“-1”** means there is no restriction.

**STEP6:** Fit the model with train data

**STEP7:** Predict and evaluate test data generated in STEP4

Both Random Forest Classifier model’s metrics are explained in section-3 evaluation below.

### 2.2.2 Pseudo code – Deep Learning – Feed Forward Neural Network:

#### 2.2.2.1 BINARY CLASSIFICATION:

**STEP 1:** Initialized Kera’s Sequential model.

**STEP 2:** Configuring model by adding different layers. (We use Dense library to build input, hidden and output layers of a neural network)  
**2.1)** The first input layer with random\_normal as kernel\_initializer with input\_dim as 39 (considering all protein biomarkers) followed by Relu activation (rectified linear activation function is easier to train and often achieves better performance) and 4 neurons.  
**2.2)** The Second dense layer is hidden layer with 4 neurons followed by Relu activation and random\_normal as kernel\_initializer(Initializers define the way to set the initial random weights of Keras layers. Here we have used random\_normal as it generates tensors with a normal distribution).  
**2.3)** Final layer with two outputs for two categories with sigmoid activation (As this is a binary classification problem, we will use sigmoid as the activation function)

**STEP 3:** Compile Model.

Figure 14 - Deep Learning model compilation attached in appendix.

**STEP 4:** Fit the Model with below hyperparameters.  
**4.1)** ADAM optimizer. (To optimize our neural network, we use Adam. Adam stands for Adaptive moment estimation. Momentum takes the past gradients into account to smooth out the gradient descent.)  
**4.2)** binary\_crossentropy cost function.  
**4.3)** Evaluation Matrix Accuracy.  
**4.4)** Training epoch with batch size as 10. (we now fit out training data to the model we created. we use a batch\_size of 10. This implies that we use 10 samples per gradient update. We iterate over 100 epochs to train the model. An epoch is an iteration over the entire data set.)  
**4.5)** Total epochs set to 100.

**STEP 5**: Training Model  
for each epoch (100):  
for each batch (1):  
Extract the batch data  
Run the optimizer + cross-entropy operations  
Add to the average cost.  
if current epoch val\_loss < epoch with least val\_loss:  
Will save current epoch. else:  
Next epoch will run on last epoch which has least validation loss.  
Calculate the current training and validation accuracy.

**STEP 6:** Evaluate Model on Test data

Deep Learning Binary Classifier model’s metrics are explained in section-3 evaluation below.

#### 2.2.2.2 MULTICLASS CLASSIFICATION:

**STEP 1:** Initialized Kera’s Sequential model.

**STEP 2:** Configuring model by adding different layers. (We use Dense library to build input, hidden and output layers of a neural network)  
**2.1)** The first input layer with random\_normal as kernel\_initializer with input\_dim as 41(considering all protein biomarkers) followed by Relu activation (rectified linear activation function is easier to train and often achieves better performance) and 8 neurons.  
**2.2)** The Second dense layer is hidden layer with 10 neurons followed by Relu activation.  
**2.3)** The Third dense layer is hidden layer with 10 neurons followed by Relu activation.

**2.4)** The Fourth dense layer is hidden layer with 10 neurons followed by Relu activation.  
**2.5)** Final layer with 8 outputs for two categories with Softmax activation (The softmax function is used as the activation function in the output layer of neural network models that predict a multinomial probability distribution)

**STEP 3:** Compile Model.

Figure 15 – Multiclass Deep Learning model compilation attached in appendix.

**STEP 4:** Fit the Model with below hyperparameters.  
**4.1)** ADAM optimizer. (To optimize our neural network, we use Adam. Adam stands for Adaptive moment estimation. Momentum takes the past gradients into account in order to smooth out the gradient descent.)  
**4.2)** categorical\_crossentropy cost function. (If it is a multiclass problem, you must use categorical\_crossentropy)  
**4.3)** Evaluation Matrix Accuracy.  
**4.4)** We iterate over 100 epochs to train the model. An epoch is an iteration over the entire data set.

**STEP 5**: Training Model  
for each epoch (100):  
for each batch (1):  
Extract the batch data  
Run the optimizer + cross-entropy operations  
Add to the average cost.  
if current epoch val\_loss < epoch with least val\_loss:  
Will save current epoch. else:  
Next epoch will run on last epoch which has least validation loss.  
Calculate the current training and validation accuracy.

**STEP 6:** Evaluate Model on Test data

Deep Learning Multi Classifier model’s metrics are explained in section-3 evaluation below.

## 2.3 Justify the Choice of the Method.

### 2.3.1 Goal

We have two types of predictions to be made for our analysis. One is the binary classification model which predicts if an individual has cancer or not (positive or negative), and the multiclassification model to localize the type of cancer.

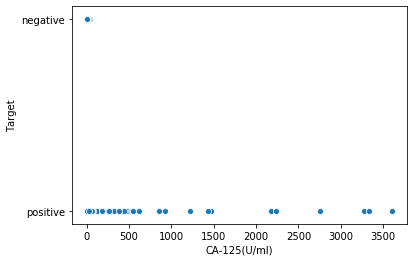
### 2.3.2 Training Process

For the binary classification model, we have trained the model on both the positive and negative individuals or classes which are almost balanced. For the multiclassification model we train only on different cancer types and skip the rows containing the samples of healthy patients. This is because when we have a strong binary classification model that predicts cancer with high accuracy, we do not have to add any further noise to the multi classification model and use the multiclassification model only when the probability of cancer is high, and to localize the type of cancer. Due to the presence of imbalanced classes in multi classification we also experimented with oversampling the data to obtain better performance.

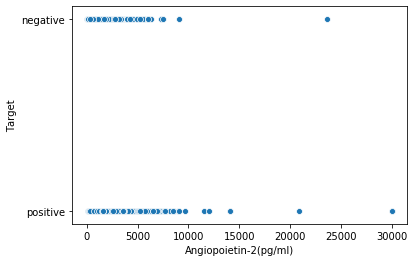
### 2.3.3 Choosing the right Model for our Data

From figure 9 and figure 10 below, when we plotted scatter plots for our dependent variable, we do not see a visible boundary line, hence instead of going with linear based models or discriminative models like logistic regression, we have chosen to approach our predictive modelling with generative modelling techniques. Similarly models like Naïve-Bayes assume that features in the training data are independent of each other which is not true of the data we have.

Hence, we have chosen a tree-based model – random forest to start out with our analysis. Since random forest generalizes better than decision tree and works well with non-linear data. Along with random forest we have also chosen deep learning due to the non-linearity in our data. ‘Sklearn’ and ‘Keras’ library help in quick prototyping of these models.



**Figure 9 – Scatter plot for Target Feature with CA-125(U/ml)**



**Figure 10 – Scatter plot for Target Feature with Angiopoietin(pg/ml)**

# 3.MODEL PERFORMANCE AND EVALUATIONS

This section explains in detail, the performance of all the experiments performed for both binary and multiclassification and the metrics on which these are compared.

## 3.1 What metrics do you use for evaluation?

Since we have built classification models, and the modelling is used on a clinical data to predict cancer, we have used sensitivity and specificity as our primary metrics to calculate the efficiency of the model. For cancer prediction both true positives and true negatives need to be predicted accurately and for the multi classification problem of localizing cancer, it is important to predict the right type of cancer. Sensitivity gives the measure of percentage of true positives predicted accurately and specificity gives the measure of percentage of true negatives predicted correctly.

## 3.2 What is your ground truth?

* For binary classification, there are two labels ‘positive’ for cancer patients and ‘negative’ for normal individuals.
* For multiclassification model there are 8 labels. ‘Breast’, ‘Colorectum’, ‘Esophagus’, ‘Liver’, ‘Lung’, ‘Ovary’, ‘Pancreas’, ‘Stomach’ – referring to 8 different types of cancer.

## 3.3 Discuss the performance and the limitation of the method.

### 3.3.1 Performance for Binary Classification

|  |  |  |
| --- | --- | --- |
| **Model** | **Sensitivity** | **Specificity** |
| **Deep Learning** | 0.8972 | 0.8425 |
| **Random forest (initial features)** | 0.9146 | 0.8947 |
| **Random forest (SFS features)** | 0.9756 | 0.9473 |
| **Decision Tree** | 0.943 | 0.9282 |

**Table 6 – Performance of different Models for Binary Classification**

From table 6 we observe that random forest model performs significantly better compared to deep learning and decision tree models. We choose initial set of features as listed in section 1.1.1 for the binary classification model. However, the random forest model was further improved in performance after we chose the feature selection process. This is explained in the next section 3.3.1.1

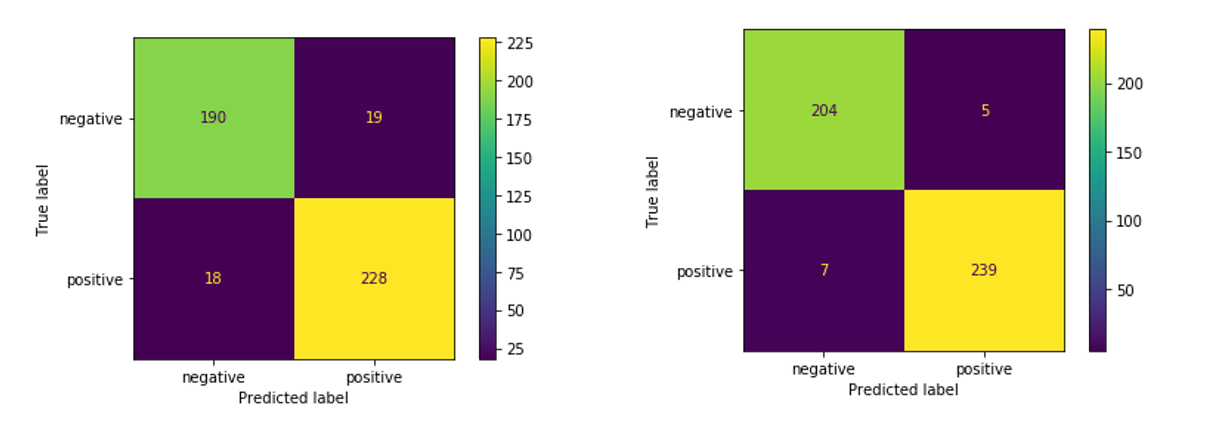
#### 3.3.1.1 Improvements for Random Forest Binary Classifier using Sequential Feature Selection

To improve the binary classification model, we approached the problem with sequential feature selection and experimented with all possible combinations of the available features to arrive at the best model. Since random forest model had given the best baseline performance, we used the random forest model with sequential feature selection and the following protein biomarkers gave the best performance.

[AFP(pg/ml), DKK1(ng/ml), G-CSF(pg/ml), GDF15(ng/ml), HE4(pg/ml), Mesothelin(ng/ml), OPG(ng/ml), sFas(pg/ml), TGFa(pg/ml)]

Unfortunately, none of the features here match the initial inputs from Table 4, that we had decided based on the boxplots, EDA and referring to the domain literature [1][2][3]. However, we have decided to include these features for our improved binary classification model. Since, we do not have a separate validation set (which is a limitation), in future when we have new data points, we plan to compare the performance using both the initial set of features from Table 4 and check if these extracted features perform as intended.

#### 3.3.1.2 Confusion Matrix for Random Forest Model Baseline Model and Improved Model



**Figure 11 – Confusion Matrix for Binary Classification using Random Forest**

From figure 11(left), if we see 19 instances of misclassification for normal healthy individuals, who are negative but have been identified as positive for cancer (False positives). And there are 18 cancer patients who have wrongly been predicted as cancer negative (False negatives). This number is reduced to 5 and 7 respectively with the improved model using feature selection.

### 3.3.2 Performance for Multi Classification

Table 7 below displays the sensitivity of each class for the 3 models tried – random forest without sampling, random forest with oversampling and deep learning model.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Classes** | **RF without sampling** | **RF with Oversampling** | **RF with hyperparameter tuning** | **Deep Learning** |
| **Breast** | 0.807692 | 0.911765 | 0.892156863 | 0.987603 |
| **Colorectum** | 0.948454 | 0.798077 | 0.826923077 | 0.93 |
| **Esophagus** | 0 | 1 | 1 | 0.987705 |
| **Liver** | 0.333333 | 1 | 1 | 0.9819 |
| **Lung** | 0.25 | 1 | 1 | 0.780645 |
| **Ovary** | 0.875 | 1 | 1 | 0.924107 |
| **Pancreas** | 0.709677 | 1 | 1 | 0.983539 |
| **Stomach** | 0 | 1 | 1 | 0.965957 |

**Table 7 – Sensitivity for different Multiclassification Models**

Table 8 below displays the specificity of each class for the 3 models tried – random forest without sampling, random forest with oversampling and deep learning model.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Classes** | **RF without sampling** | **RF with Oversampling** | **RF with Hyperparameter Tuning** | **Deep Learning** |
| **Breast** | 0.915 | 0.991098 | 0.995548961 | 0 |
| **Colorectum** | 0.651613 | 0.994048 | 0.989583333 | 0.807692 |
| **Esophagus** | 1 | 1 | 0.998496241 | 0.625 |
| **Liver** | 1 | 0.997114 | 0.998556999 | 0.645161 |
| **Lung** | 0.977679 | 0.985653 | 0.987087518 | 0.773196 |
| **Ovary** | 0.995902 | 1 | 1 | 0.607143 |
| **Pancreas** | 0.99095 | 0.995575 | 0.994100295 | 0.333333 |
| **Stomach** | 1 | 0.992582 | 0.986646884 | 0.176471 |

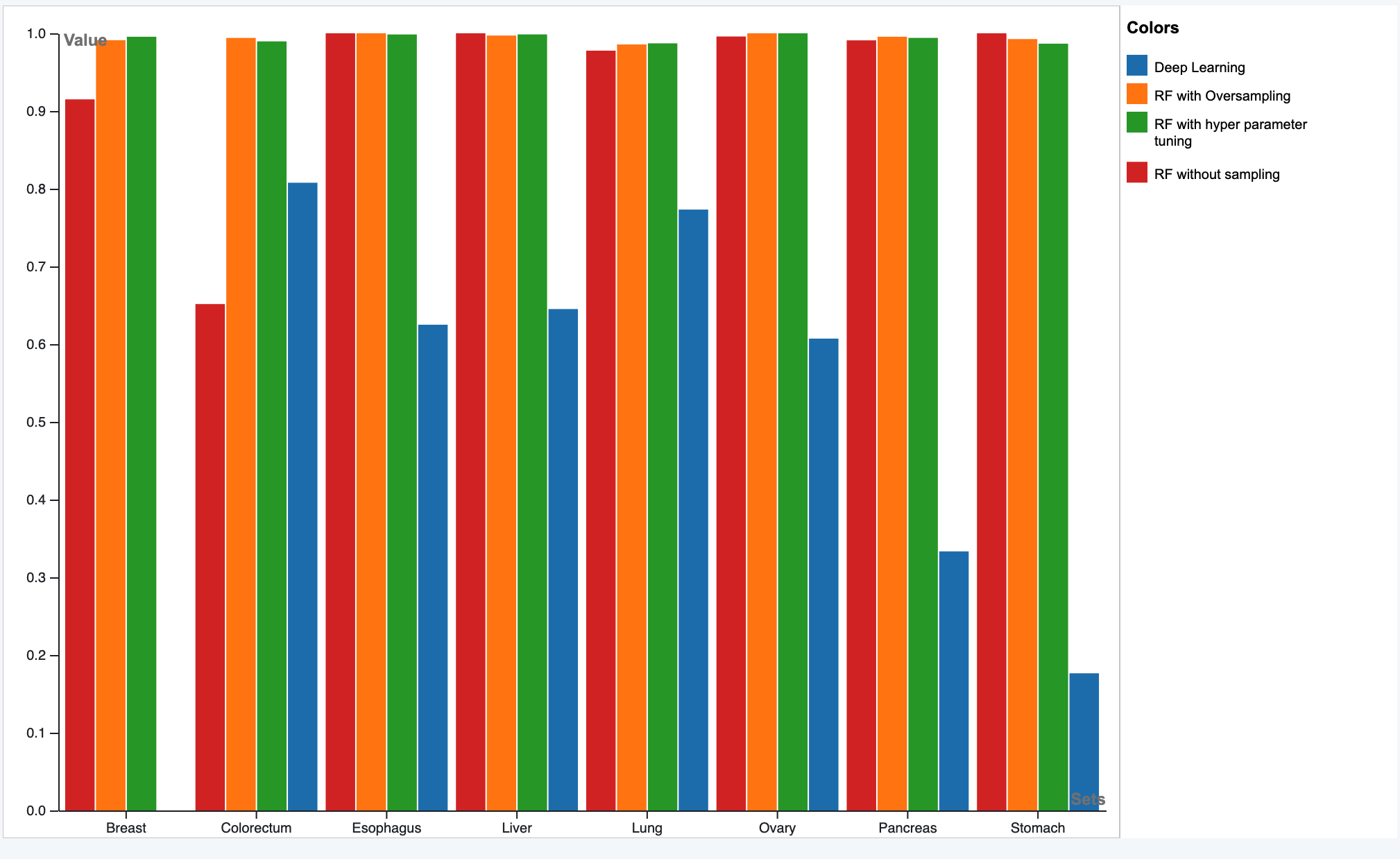
**Table 8 – Specificity for different Multiclassification Models**

The figure 11 below summarizes and compares all the multiclassification models for sensitivity and figure 12 for specificity, respectively.

Chart, bar chart

Description automatically generated

**Figure 11 – Comparing Sensitivity of different models**



**Figure 12 – Comparing Specificity of different models**

#### 3.3.2.1 Performance Comparison for Random Forest with sampling, without sampling and Deep Learning

From the above two graphs and Table 7 and 8, we see that the quality of predictions for **random forest** has improved significantly after oversampling our data and balancing the classes, especially the sensitivity for liver, esophagus, lung, and liver cancer have improved for the random forest model with oversampled data. However, we see a slight drop in sensitivity for ‘colorectum’ and ‘breast’ cancer after oversampling the data. The sensitivity for colorectum is improved further after hyperparameter tuning but the sensitivity for breast cancer is slightly dropped. We noticed that the hyperparameters were majorly sensitive to two classes ‘Colorectum’ and ‘Breast’ whereas the sensitivity and specificity of other classes did not change as much while compared to the random forest model with oversampling. This can be seen in Figures 11, 12 and Table 7 and 8.

The **deep learning model** has done well as is, on the unsampled data, with respect to predicting true positives for each cancer type, we see high values of sensitivity even for colorectum and breast cancer beating the random forest model. But the model has done poorly in identifying the true negatives and hence the specificity values are low especially for breast, pancreas, and stomach cancer. This is due to the presence of increased false positives. Perhaps, deep learning could improve in its performance if we can get more training data.

Overall, the random forest algorithm is working well for multiclassification and we tried to improve it with hyperparameter tuning.

#### 3.3.2.2 Hyperparameter Tuning for Random Forest Model

The random forest model after oversampling the data showed improved performance as compared to the performance when using imbalanced classes. However, there were few misclassifications for colorectum, and breast cancer as shown by their lower sensitivity values. We attempted hyperparameter tuning using ‘GridSearchCV’ to arrive at optimal parameters for the model by varying the number of estimators (n\_estimators), minimum number of samples required to split an internal node (min\_samples\_split) and maximum depth of each tree in the forest(max\_depth). The baseline model had n\_estimators = 100, min\_samples\_split = 2 and max\_depth = None (which expands nodes until all leaves contain less than min\_samples\_split). The average depth of the decision tree of resulting model was around ‘18’. Based on these values we tried to design the grid search parameter grid as follows -

param\_grid = {  
'bootstrap': [True],  
'max\_depth': [10, 15, 16, 17, 18, 19, 20,25,30],  
'min\_samples\_split': [2, 3, 4, 5, 6],  
'n\_estimators': [100, 150, 200, 250, 300]  
}

The following parameters gave the best results

[max\_depth=18, min\_samples\_leaf=1, min\_samples\_split=4, n\_estimators=250]

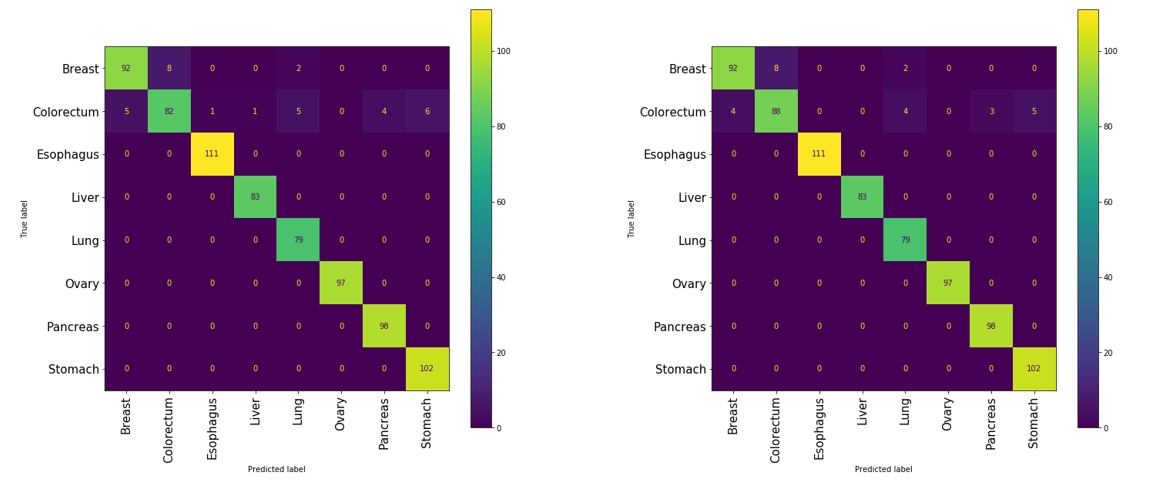
We noticed that the hyperparameters were majorly sensitive to two classes ‘Colorectum’ and ‘Breast’ whereas the sensitivity and specificity of other classes did not change as much while compared to the baseline random forest model with oversampling. This can be seen in Figures 11, 12 and Table 7 and 8.

#### 3.3.2.3 Performance Comparison of Random Forest Model with and without Hyperparameter Tuning

From the confusion matrix in figure 13(left), for random forest with oversampling, we see that while predicting ‘breast’ cancer there are quite a few false negatives and false positives. The model has mislabeled 5 instances each as ‘colorectum’ and ‘lung’ cancer instead of ‘Breast’ cancer (False negatives). It has also misclassified 8 instances which were for ‘colorectum’ cancer as ‘Breast’ cancer.

Going by the second row, we see many instances of ‘Colorectum’ cancer being misclassified into other types of cancer predominantly stomach, lung, and pancreatic cancer.

After tuning the model as per the hyperparameters mentioned in section 3.3.2.1, we noticed that true positives for ‘colorectum’ cancer increased to 88 from 82 (second row of confusion matrix) improving the sensitivity. However, we did not see any improvement with respect to those ‘8’ instances of breast cancer which are still predicted as ‘Colorectum’ cancer (as shown by first row, second column).



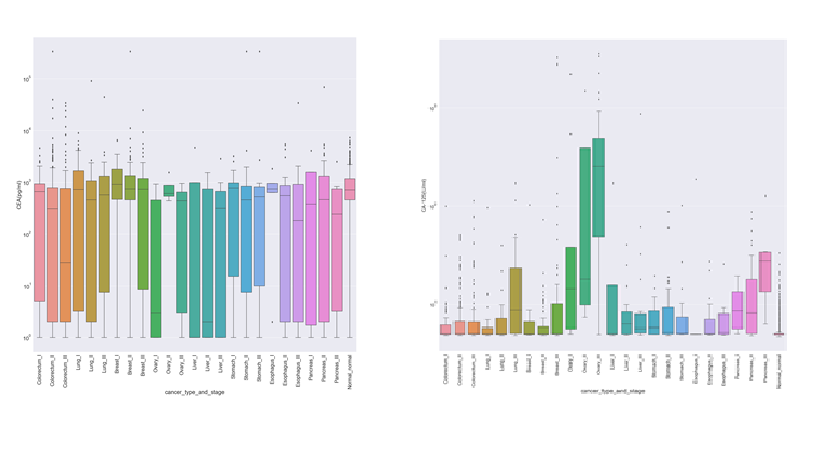
**Figure 13 – Confusion matrix for random forest with oversampling**

### 3.4 Diagnostics and Limitations

Random forest with oversampling has given us favorable results and we tried to improve it further by using hyperparameter tuning.

However even with hyperparameter tuning we noticed that it is difficult to improve sensitivity for breast and colorectum cancer at the same time and one set of parameters can improve the sensitivity for one class while it slightly drops for another. Hence, we can infer that the current model does not classify all instances of colorectum and breast cancer correctly affecting their sensitivity.

The Carcino Embryonic Antigen (CEA) is said to be a key protein biomarker for colorectum and breast cancer, but the distribution of this feature does not distinctly separate or help identify these two classes as shown in figure 14. Whereas the concentration of key protein biomarkers for other types of cancer has a distinct distribution, for example CA-125 is a key protein for detecting ‘Ovary’ cancer and from the boxplot distribution of this protein against different types of cancer – it can be clearly inferred that for ovary cancer, the concentration levels is higher. This could be one of the reasons why the sensitivity of other cancer type is higher but hard to improve for colorectum and breast cancer with the current data. Perhaps, getting more data for both these cancer types could help us.



**Figure 14 – Distribution of CEA (left) and CA-125 (right)**

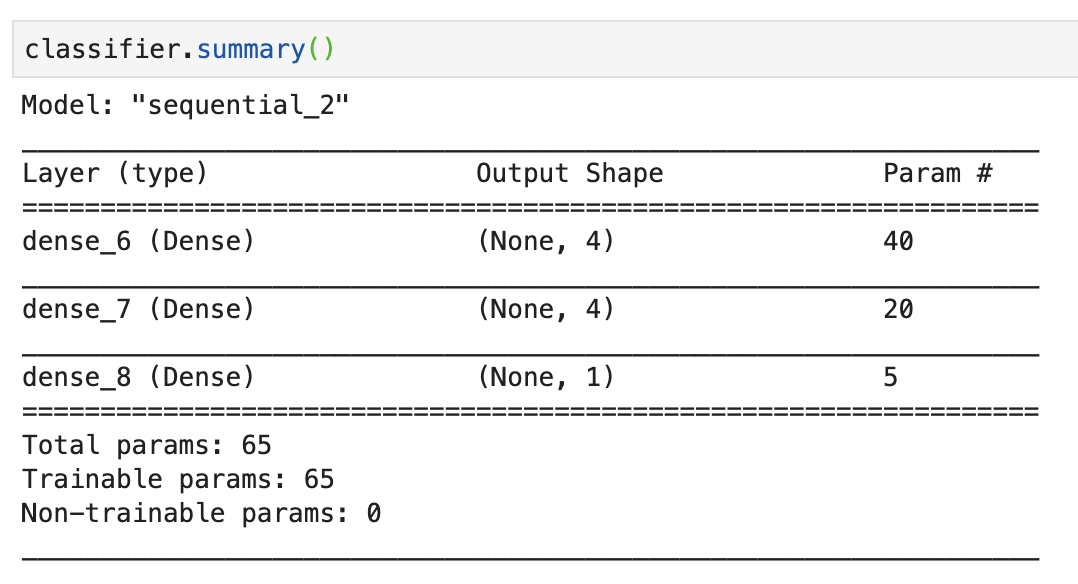
# 4.CONCLUSION

We conducted various experiments for our problem based on nonlinear classification models for both binary and multiclassification. Given the non – linear nature of the cancer protein data with respect to target classes we found that random forest model fared well in both binary and multiclassification problems. In the presence of a true validation set the model could have been validated better. The multiclassification model can further be improved for colorectum and breast cancer with more data points or if the key protein biomarkers (like CEA) display a distinct distribution for these cancer types as compared to others. We also see potential in the deep learning model if more data is available since it identified true positives for various cancer types even with imbalanced classes and limited data. However, the explain ability of the model can be a challenge with deep learning. Probabilistic and tree-based models like Naïve-Bayes, Decision trees, random forest if performed better can be more ideal and safer to be used for clinical data.

# 5.APPENDIX

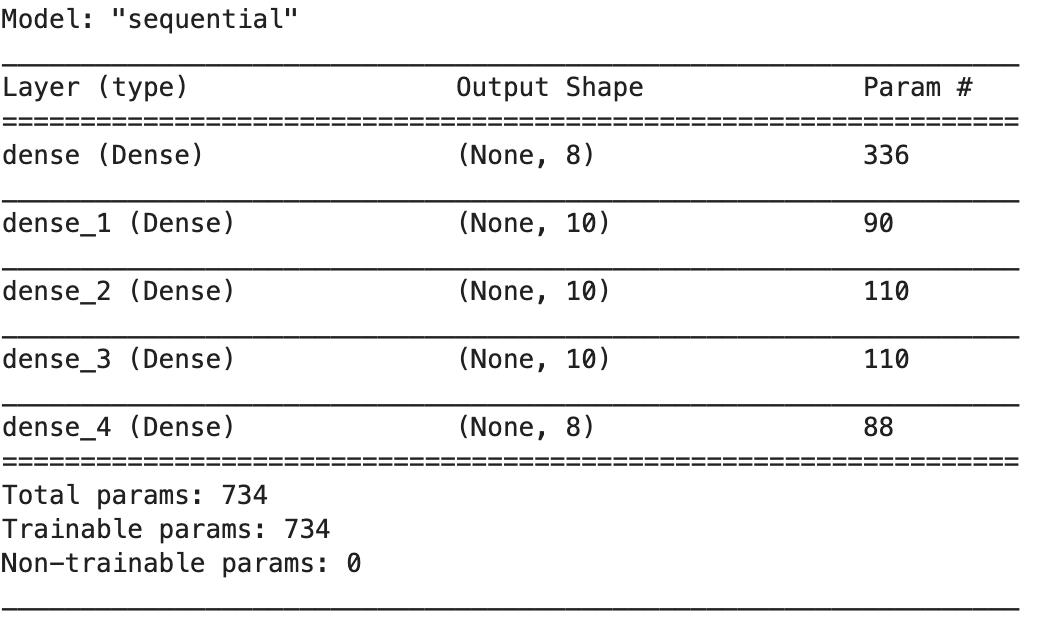
[Addition materials that are not included in the above sections.]

## 4.1 Feed forward Neural network binary Classifier using Keras



**Figure 15 – Binary Deep Learning model compilation**

## 4.2 Feed forward Neural network binary Classifier using Keras



**Figure 16 – Multiclass Deep Learning model compilation**

# 5. REFERENCES

[1] Ka-Chun Wong, Junyi Chen, Jiao Zhang, Jiecong Lin, Shankai Yan, Shxiong Zhang, Xiangtao Li, Cheng Liang, Chengbin Peng, Qiuzhen Lin, Sam Kwong, Jun Yu, Early Cancer Detection from Multianalyte Blood Test Results, Science (<https://www.sciencedirect.com/science/article/pii/S2589004219301324>)

[2] Cohen, J., Li, L., Wang, Y., Thoburn, C., Afsari, B., Danilova, L., Douville, C., Javed, A., Wong, F., Mattox, A., Hruban, R., Wolfgang, C., Goggins, M., Dal Molin, M., Wang, T.L., Roden, R., Klein, A., Ptak, J., Dobbyn, L., Schaefer, J., Silliman, N., Popoli, M., Vogelstein, J., Browne, J., Schoen, R., Brand, R., Tie, J., Gibbs, P., Wong, H.L., Mansfield, A., Jen, J., Hanash, S., Falconi, M., Allen, P., Zhou, S., Bettegowda, C., Diaz, L., Tomasetti, C., Kinzler, K., Vogelstein, B., Lennon, A., & Papadopoulos, N. (2018). Detection and localization of surgically resectable cancers with a multi-analyte blood test*. Science, 359(6378), 926–930.*

[3] Gam LH. Breast cancer and protein biomarkers. *World J Exp Med*. 2012;2(5):86-91. Published 2012 Oct 20. doi:10.5493/wjem.v2.i5.86

[4] <https://www.datacamp.com/community/tutorials/random-forests-classifier-python>

[5] <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html>

[6] <https://dataaspirant.com/random-forest-algorithm-machine-learing/>

[7] <https://towardsdatascience.com/deep-learning-feedforward-neural-network-26a6705dbdc7>

Table of Contributions

The table below identifies contributors to various sections of this document.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Section** | **Writing** | **Editing** |
| **1** | **Predictive Modeling Problem Definition** | **Ryner/Astha** | **Divya/Sudharshan** |
| **2** | **Predictive Models** | **Sudharshan/Divya** | **Madhu/Astha** |
| **3** | **Evaluations** | **Madhu/Divya** | **Ryner/Astha** |
| **4** | **Appendix** |  |  |

**Grading**

The grade is given on the basis of quality, clarity, presentation, completeness, and writing of each section in the report. This is the grade of the group. Individual grades will be assigned at the end of the term when peer reviews are collected.