**Breast Cancer Survival Prediction**

**Premise**

The most important information that usually patients diagnosed with breast cancer want to know is how long they will survive. Patients with different criteria when they have been digones can have very different survival rates. The intention of this study is to predict survival rate for patients with different diagnostic criteria.

**Dataset**

Dataset that has been used here is [Seer Breast Cancer Data](https://ieee-dataport.org/open-access/seer-breast-cancer-data#files), This dataset of breast cancer patients was obtained from the 2017 November update of the SEER Program of the NCI, which provides information on population-based cancer statistics. The dataset involved female patients with infiltrating duct and lobular carcinoma breast cancer diagnosed in 2006-2010.

This project was done in 4 main steps:

1. **Data Wrangling** - This is where we try to get a raw understanding of what the data looks like and perform transformations and operations on it to bring it to a more usable state
2. **Exploratory Data Analysis** - Here, we try to understand the nature and distribution of our data by plotting visualizations of different features in the dataset. This can give useful insights prior to the modeling phase.
3. **Preprocessing** - The stage where we perform any other operations on the data, like scaling or normalizing, before it is ready to be fed into a machine learning model.
4. **Modeling** - The most important part of course, where we actually try fitting different models onto the preprocessed dataset and evaluate which model performs the best

**Data Wrangling**

The good thing is, this dataset is clean but still needs some extra clean up and preprocessing before it's ready to be fitted to a model.

Each sample of data consists of age, race, marital status, t stage, n stage,a stage, tumor size, estrogen status, progesterone status,regional nodes examined, regional nodes positive and status of the patient.

So clearly, this is a very rich dataset and there is plenty of relevant information our models can use while training. However, the challenge is The dataset has a few features that are continuous and the rest are categorical variables.

**1.Data Cleaning**

As investigated there are 10 categorical columns and 5 numerical columns. Also there seems to be no null value for all the records.

Another step to clean the data is removing the unnecessary text from the records. For example, the “Grade” feature has the below records:

1. 'Moderately differentiated; Grade II'
2. 'Poorly differentiated; Grade III'
3. 'Well differentiated; Grade I'
4. 'Undifferentiated; anaplastic; Grade IV'

We get enough information by just having the grades as below:

1. Grade II
2. Grade III
3. Grade I
4. Grade IV

To be able to do proper statistical analysis we are going to convert the categorical columns to numeric data

**2.Convert Data to Dummy Data**

Used two methods to convert the data to numeric in this step. For the features “one hot encoding” was used as below with dropped one column to avoid Dummy Variable Trap and multicollinearity.

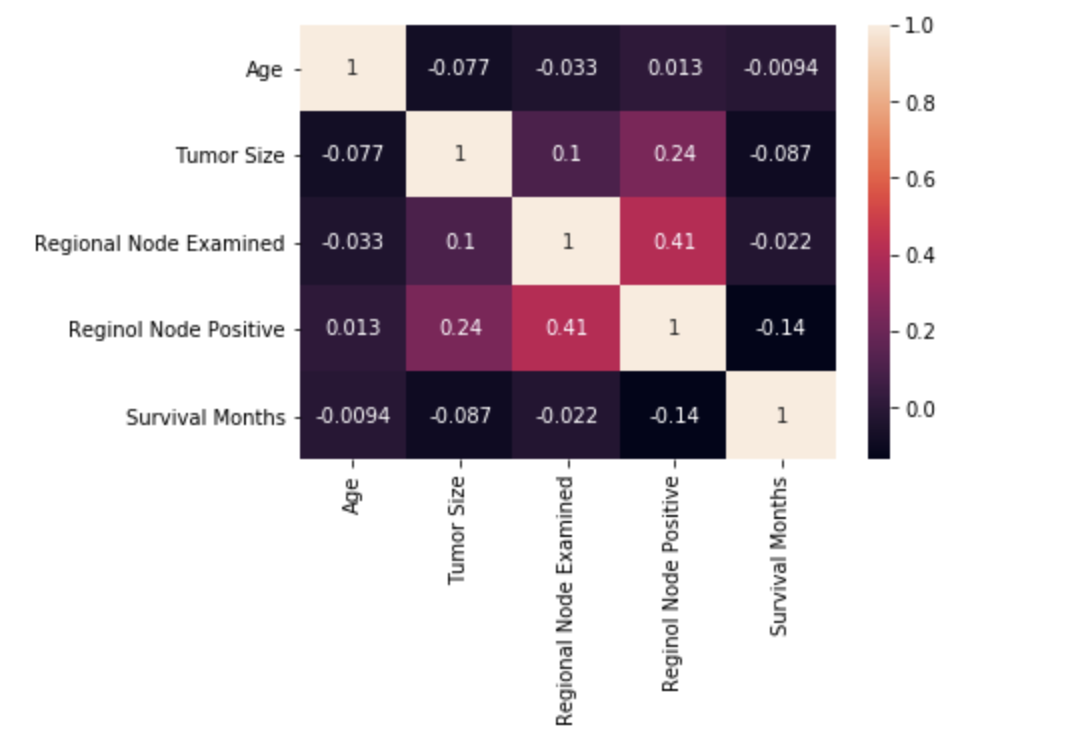
* features = pd.get\_dummies(features,drop\_first = True)

Encoded the target column using LabelEncoder as below:

* target **=** pd**.**DataFrame(le**.**fit\_transform(target),columns **=** ['Status'])

**3.Correlation between features and target**

To visualize the correlation between the continuous features, Heatmap was used as below:

****

As seen in the above map there are some correlations between the features such as Regional Node Examined with Regional Node Positive and Tumor size. There will be deeper investigation in the next section.

**4.Balancing the Data**

Imbalanced classification is the problem of classification when there is an unequal distribution of classes in the training dataset. Let's investigate how our target data look:

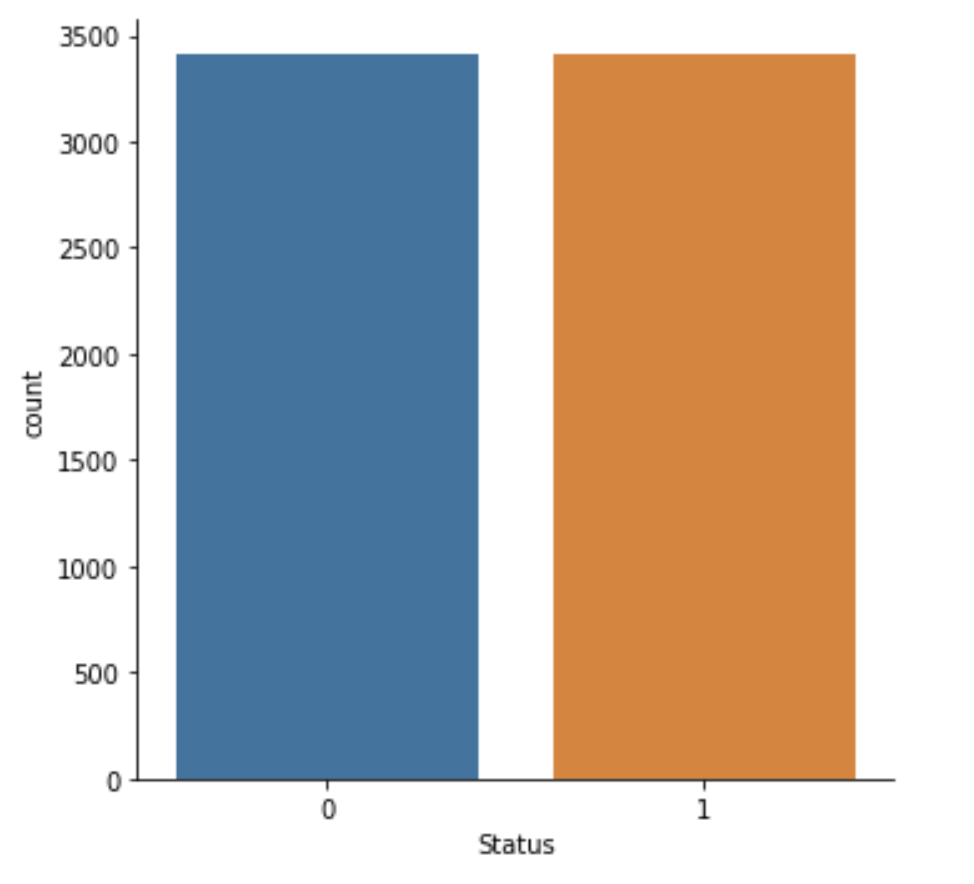
sns**.**catplot(x**=**'Status',data**=**y\_test,datakind**=**"count");



Obviously our target data is not balanced and the number of alive records are almost 5th times the records with dead status. To balance the data SMOTE technique was used:

* oversample **=** SMOTE()
* X\_smote, y\_smote **=** oversample**.**fit\_resample(X, y)
* sns**.**catplot(x**=**'Status',data**=**y\_smote,kind**=**"count");

After resampling(over sample) our data, records are balanced and the targets are equally disputed between Live and Dead:

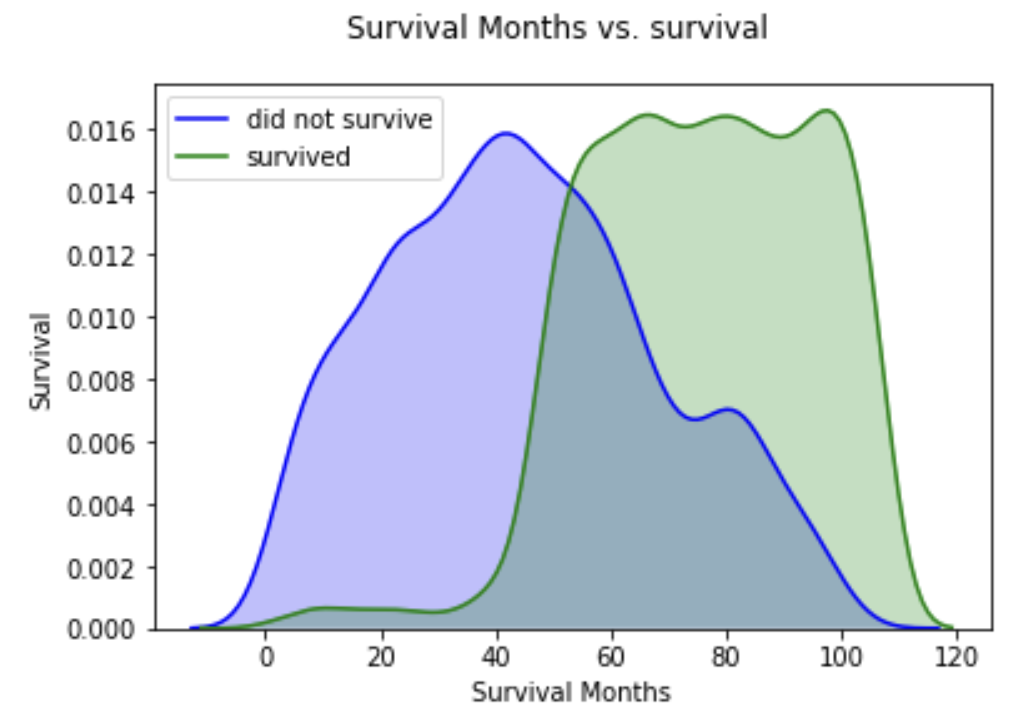
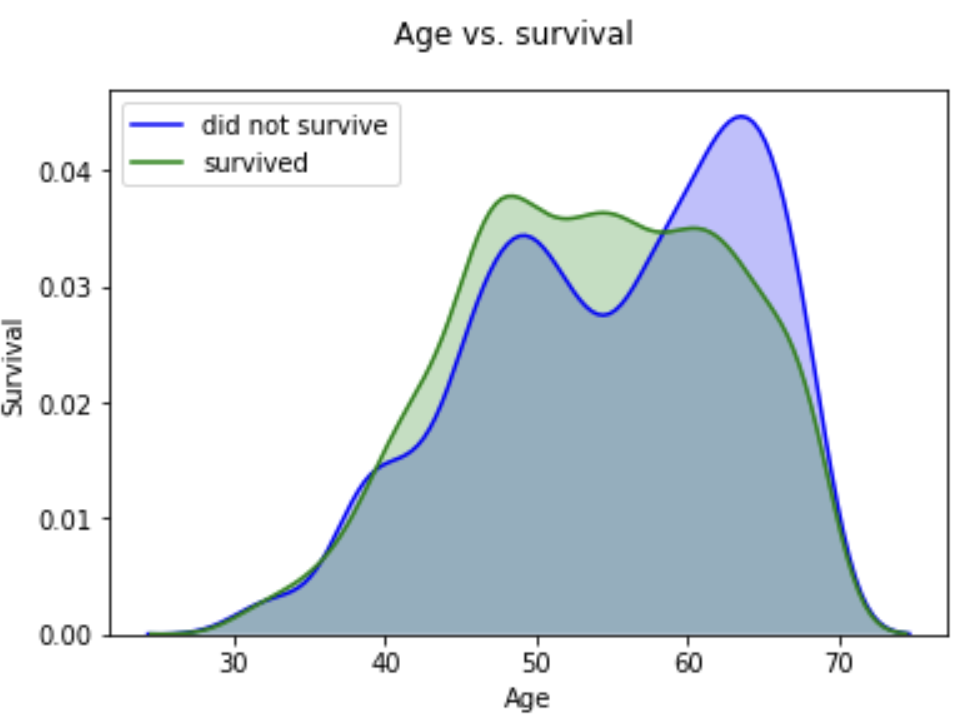


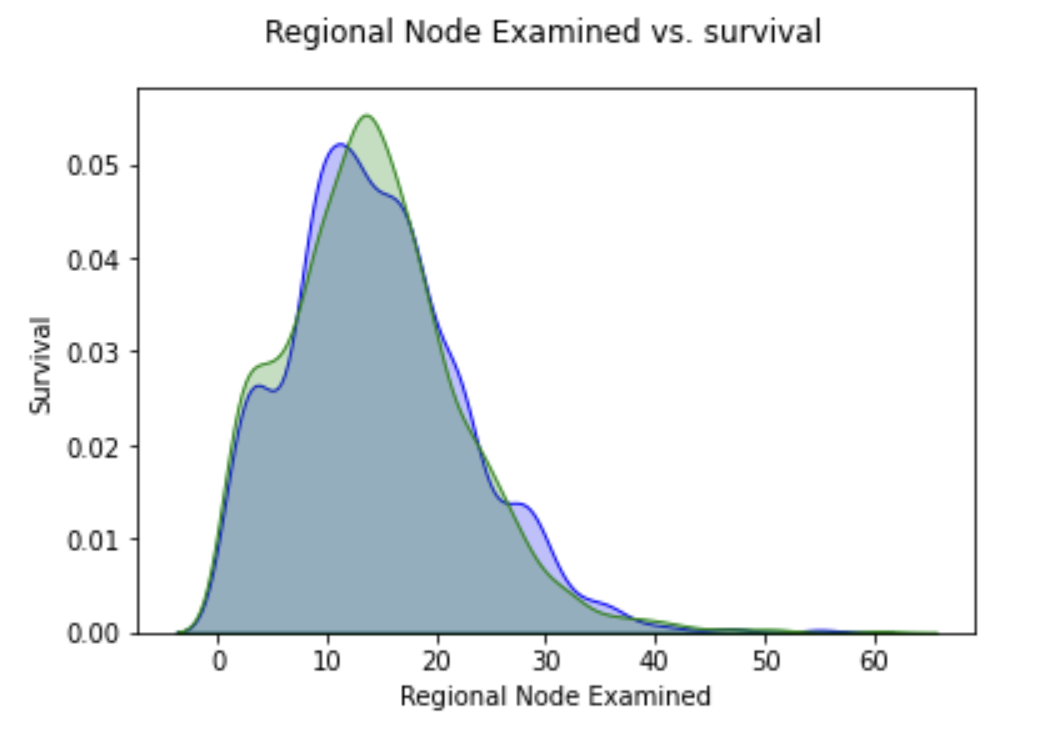
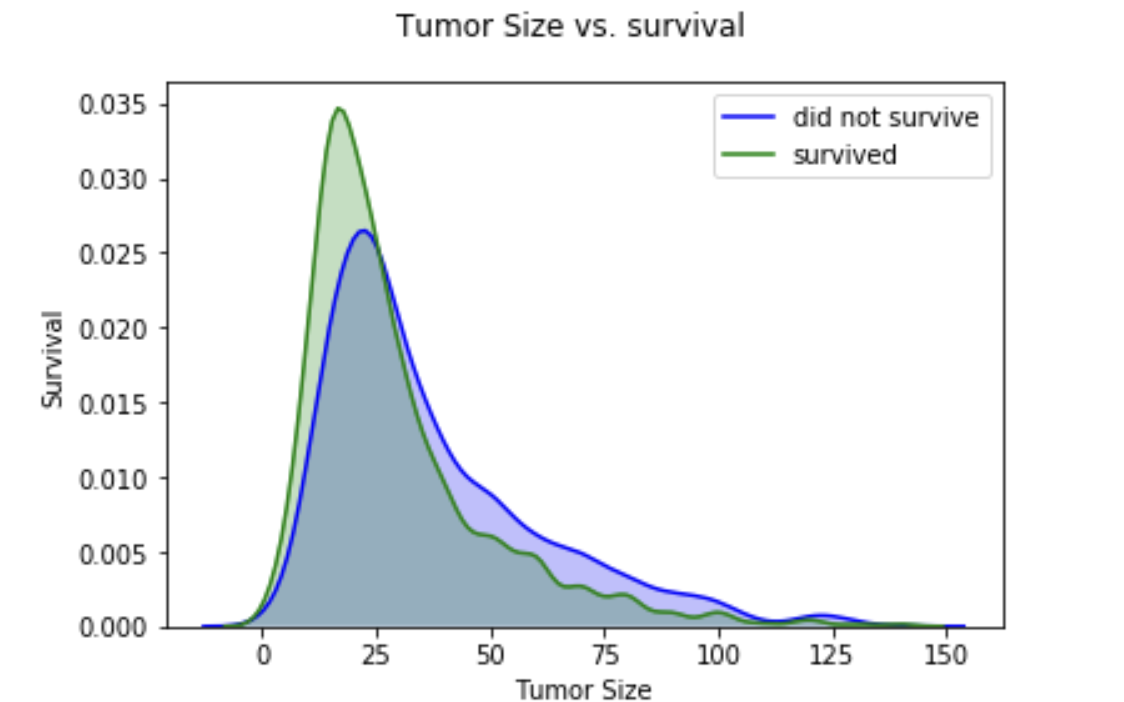
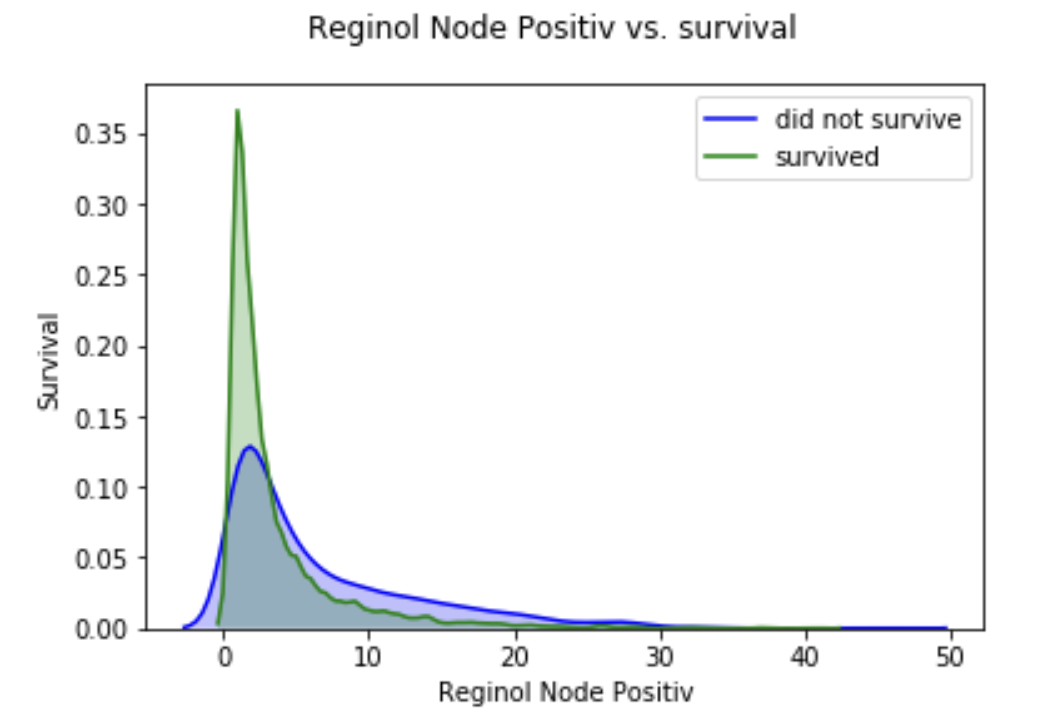
**Exploratory Data Analysis**

In this section we will dive deeper to explore the data. We will explore the correlation of continuous features and the relationship of the categorial features with the target.

1. **Continuous data**

There are 5 continuous features including "Tumor Size","Regional Node Positive"'Regional Node Examined',"Age" and ”Survival Months” . As the target is still categorical we used sns**.**kdeplot to plot each feature against the values of the target.

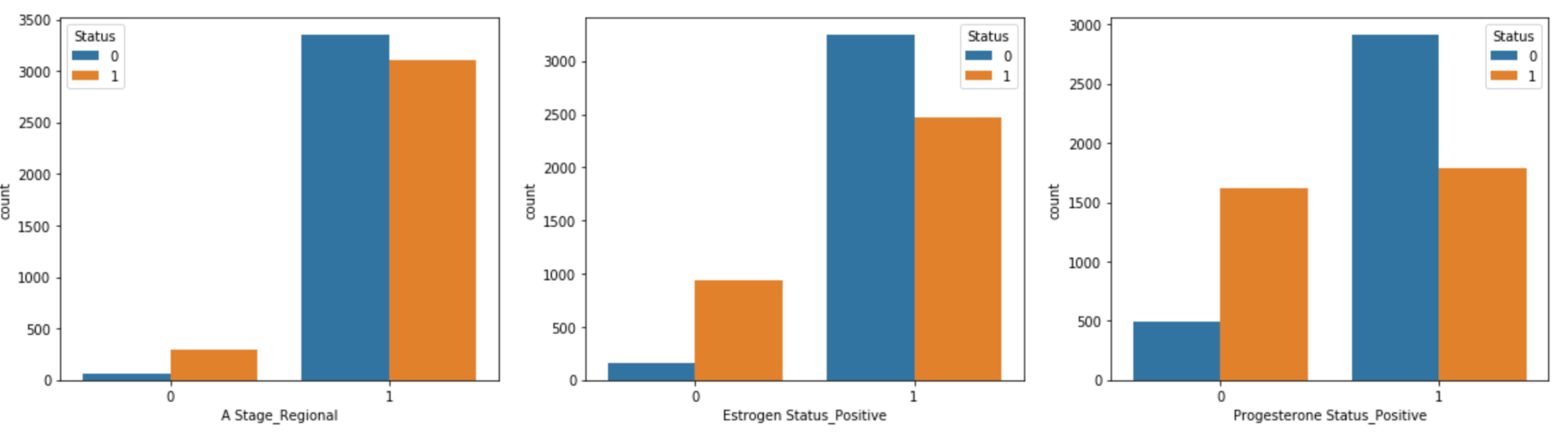
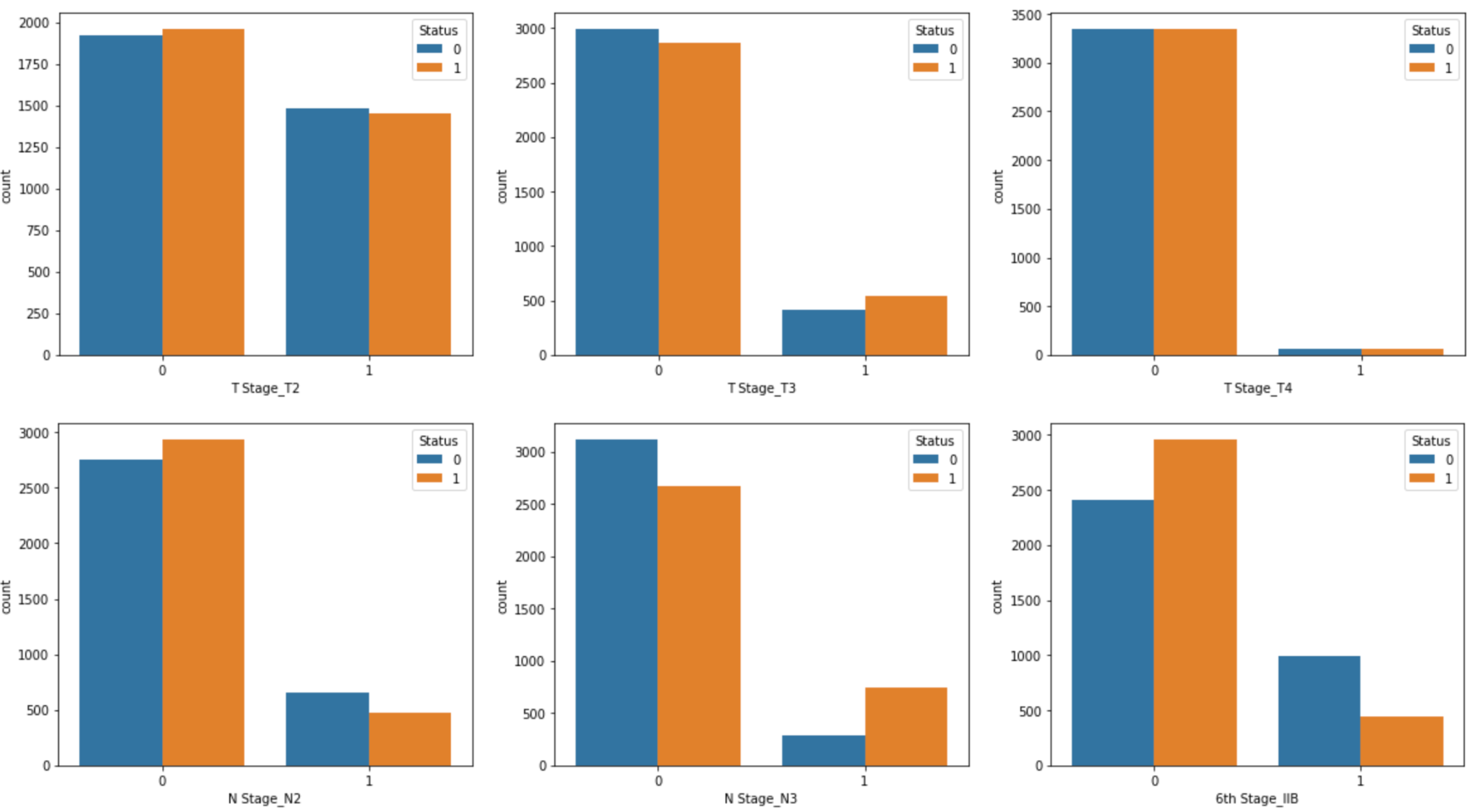
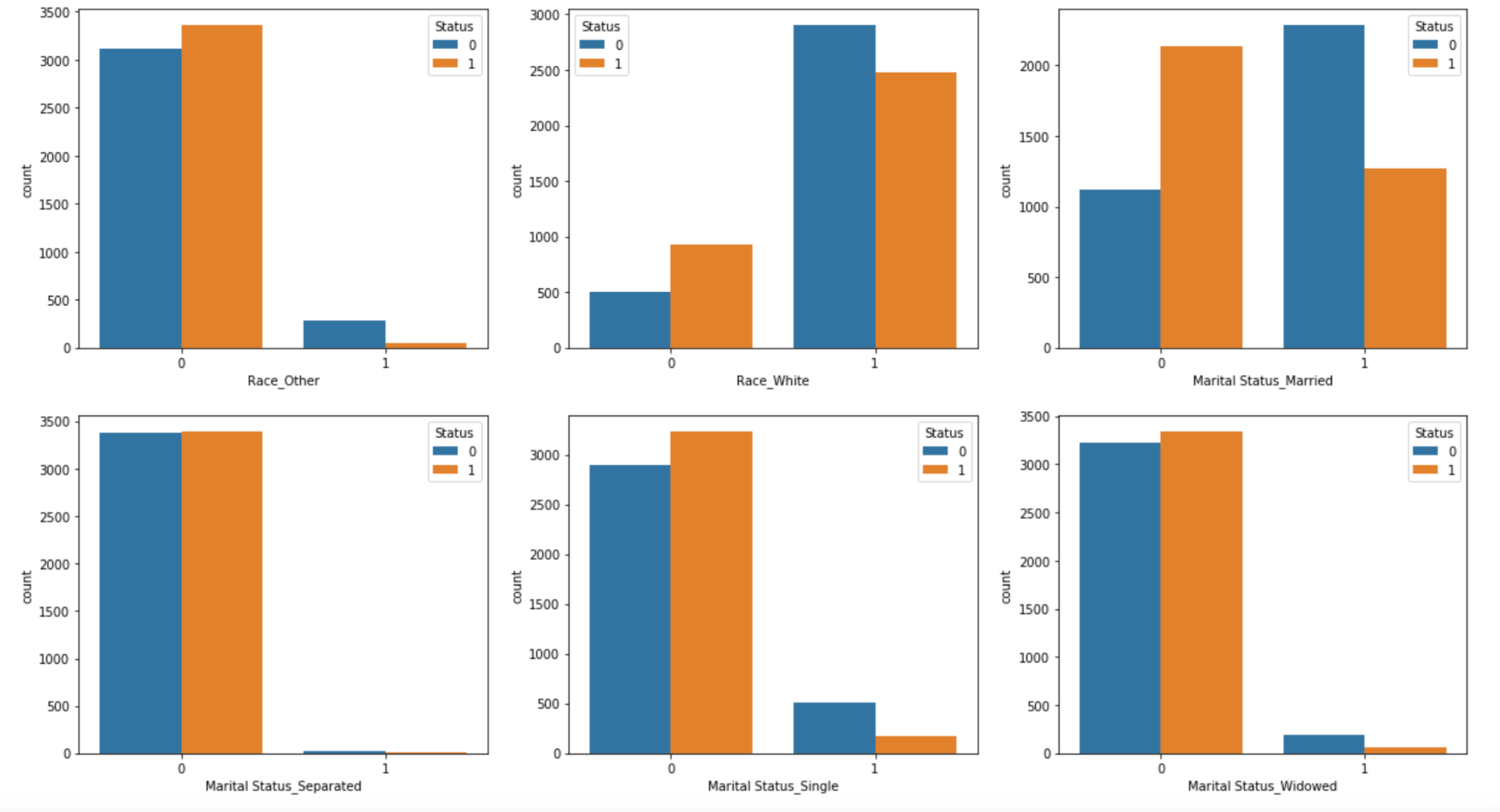




As seen in the plots, there is not an obvious correlation or relation between the features and the target except for “Survival Months”, which obviously is longer for surviving patients and much less for the dead patients.

1. **Categorical data**

The rest of the features are categorical, to compare the relationship we plot the count bars for each target value.



As seen in the above graphs some features show some contributions to the target classification. For example, married people could live longer, or people with grade 2 cancer live longer and that could be that their grade is higher and they will not survive so long. Also Being Estrogen or Progesterone positive could lead to a longer life.

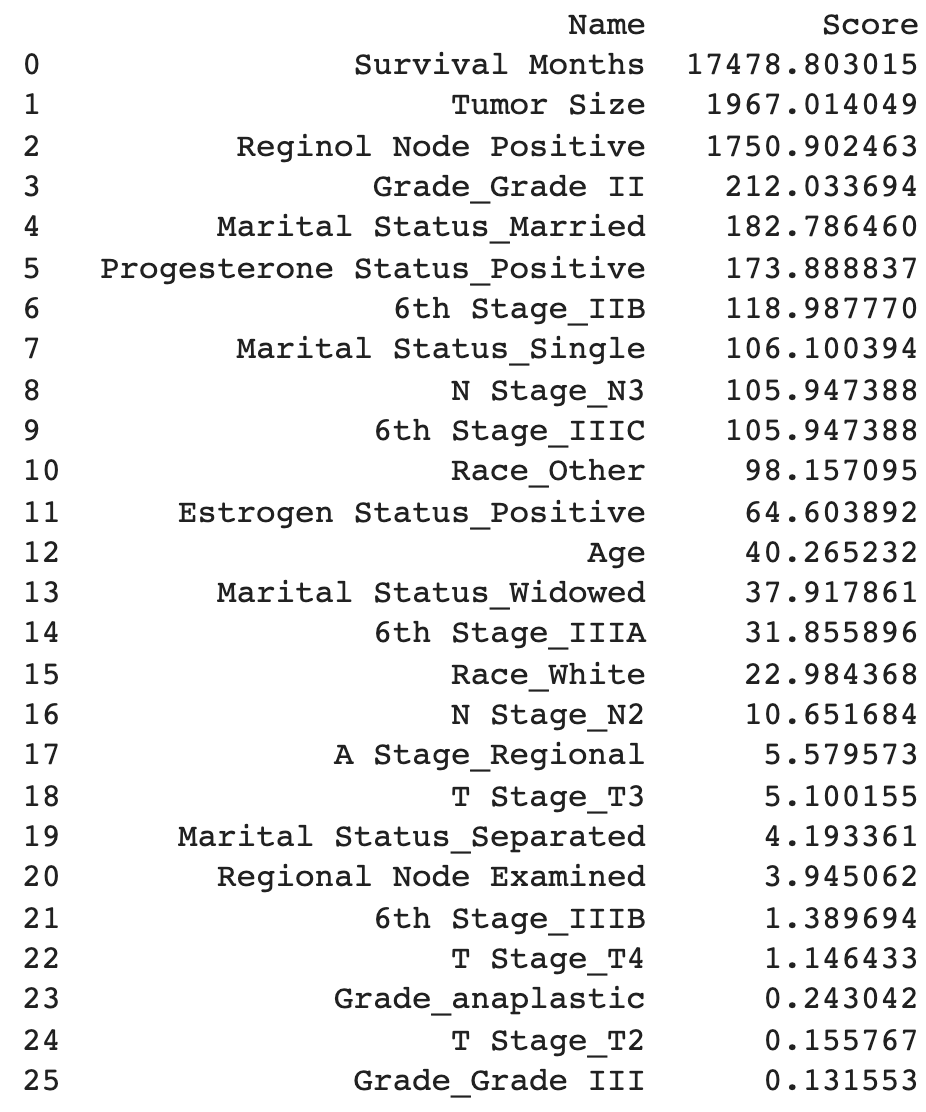
**Preprocessing**

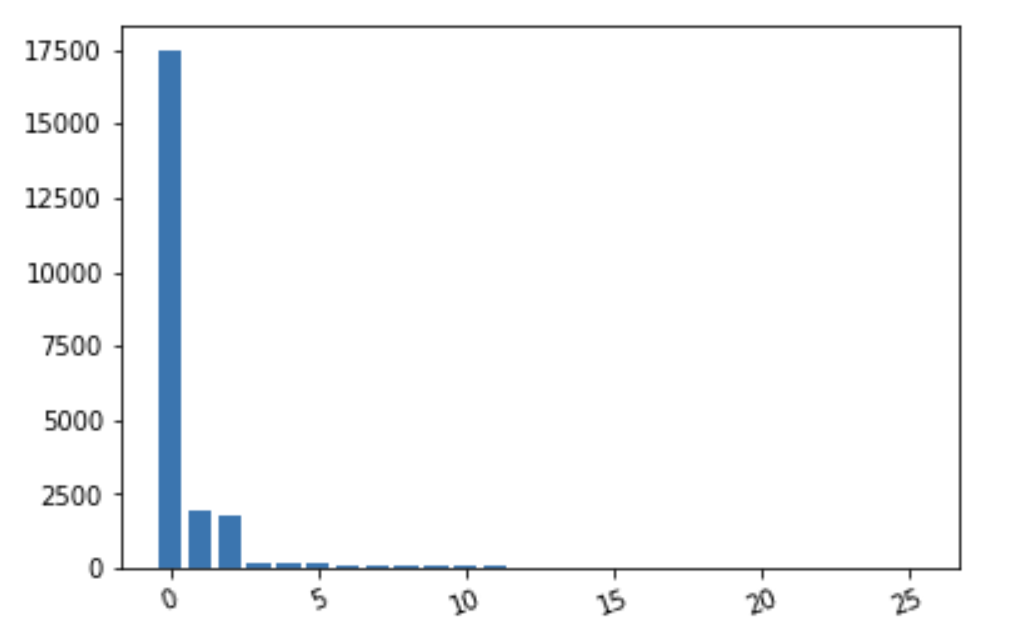
# **1.Feature Selection**

There are two popular feature selection techniques that can be used for categorical input data and a categorical target variable, which are Chi-Squared Statistic and Mutual Information Statistic.

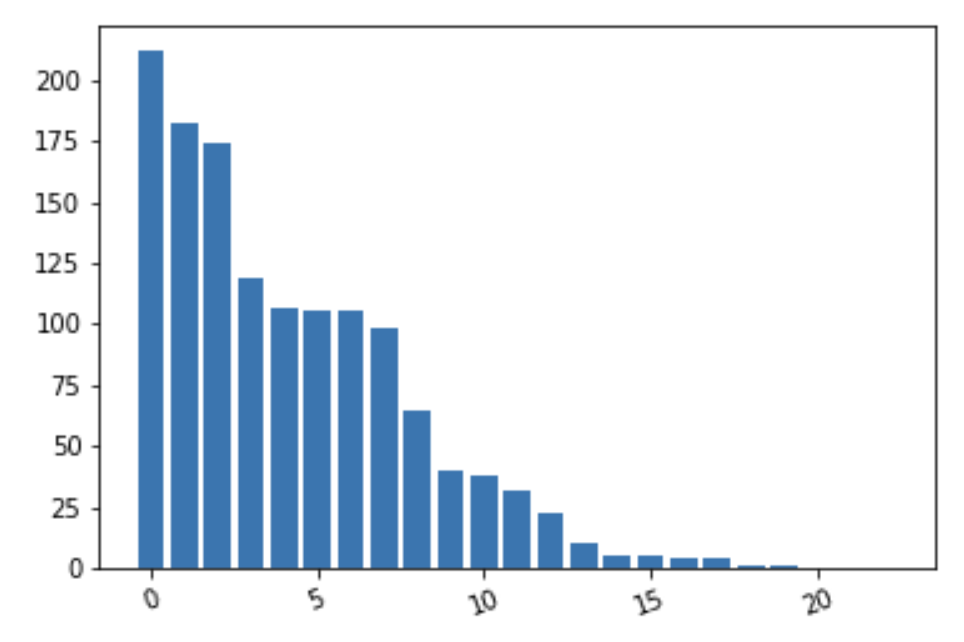
We will use the Chi-Squared Statistic.

* fs **=** SelectKBest(score\_func**=**chi2, k**=**'all')
* fs**.**fit(X\_train, y\_train)





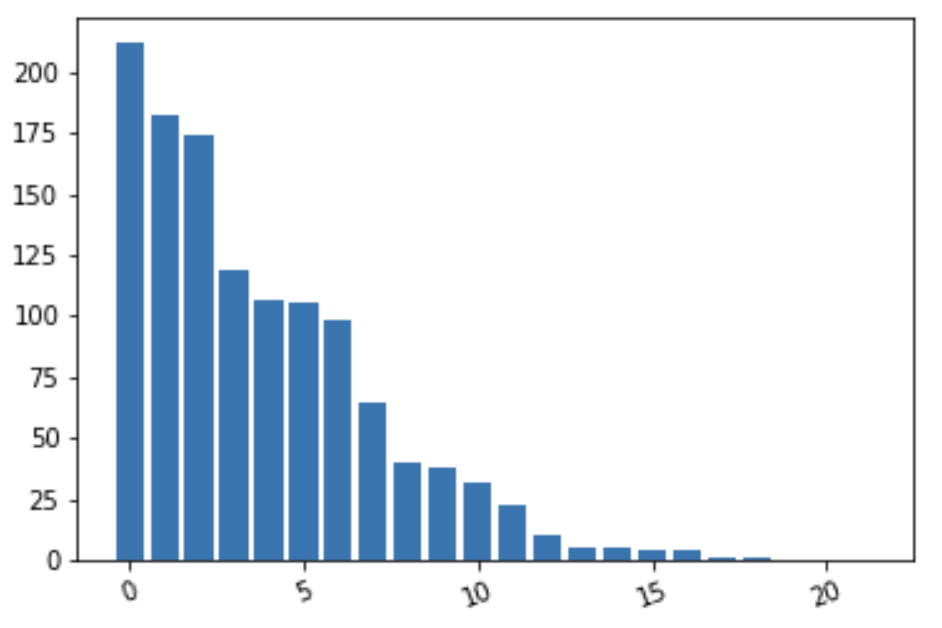
Let's drop survival month and tumor size and positive node as they have very high impact on the target data and explore the rest features



As seen above these features are the most important ones according to our KBest model:

1. Grade\_Grade II
2. Progesterone Status\_Positive
3. N Stage\_N3
4. 6th Stage\_IIIC
5. Marital Status\_Married

As discovered in data wrangling T stage,N stage and 6th stage had correlation with each other. Therefore we can just select either N Stage\_N3, 6th Stage\_IIIC and drop one. I will drop N Stage\_N3.



As seen above these features are the most important ones according to our new KBest model:

1. Survival Months
2. Tumor Size
3. Regional Node Positive
4. Grade\_Grade II
5. Progesterone Status\_Positive
6. 6th Stage\_IIIC
7. Marital Status\_Married
8. Marital Status\_Single
9. Race\_Other
10. Estrogen Status\_Positive

# **2. Split the data to test and train**

For the step which is fitting the data to the models, we need to split the data to train and test as below:

* X\_train, X\_test, y\_train, y\_test **=** prepare\_train\_test(df,'Status')
* print('X train shape is:{}'**.**format(X\_train**.**shape))
* print('y train shape is:{}'**.**format(y\_train**.**shape))
* print('X test shape is:{}'**.**format(X\_test**.**shape))
* print('y test shape is:{}'**.**format(y\_test**.**shape))

Also we need to scale the continuous data before applying the model.

# **3. Scale the data**

We use the MinMaxScaler to scale our data as below:

* scaler **=** MinMaxScaler(feature\_range**=**(0, 5))
* X\_train\_scaled **=** scaler**.**fit\_transform(X\_train)
* X\_test\_scaled **=** scaler**.**transform(X\_test)

After scaling the data , we will apply the same steps to the selected data, which is data with top selected data,

* df\_feature\_selection **=** df[['Survival Months','Tumor Size','Reginol Node Positive','Grade\_Grade II','Progesterone Status\_Positive','6th Stage\_IIIC','Marital Status\_Married','Estrogen Status\_Positive','Status']]
* X\_train, X\_test, y\_train, y\_test **=** prepare\_train\_test(df\_feature\_selection,'Status')
* scaler **=** MinMaxScaler(feature\_range**=**(0, 5))
* X\_train\_scaled **=** scaler**.**fit\_transform(X\_train)
* X\_test\_scaled **=** scaler**.**transform(X\_test)

**Modeling**

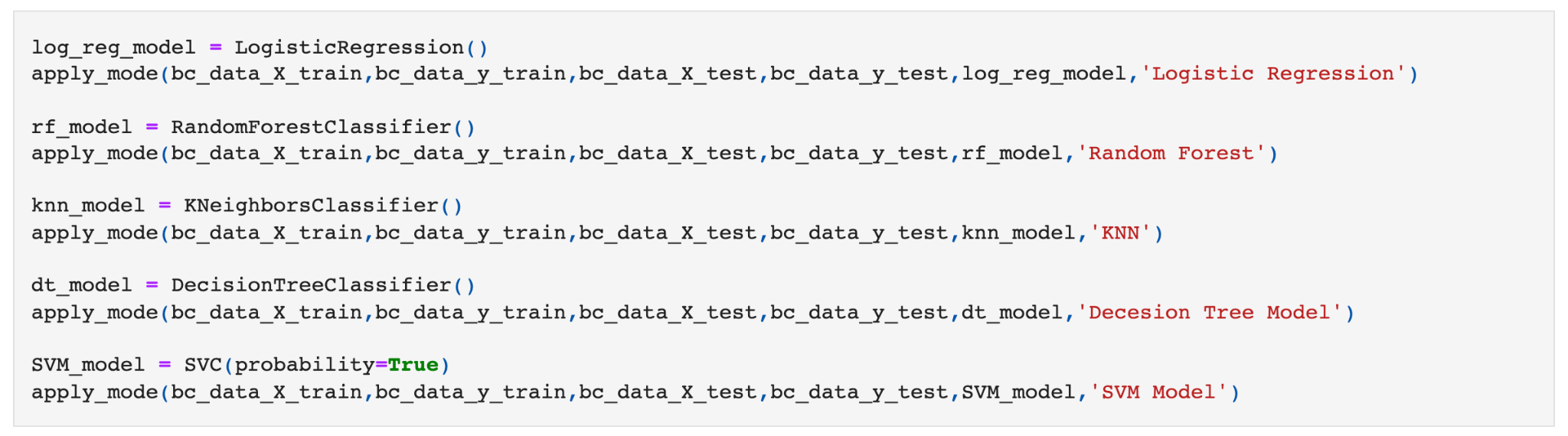
## **1.Explore different models with default values**

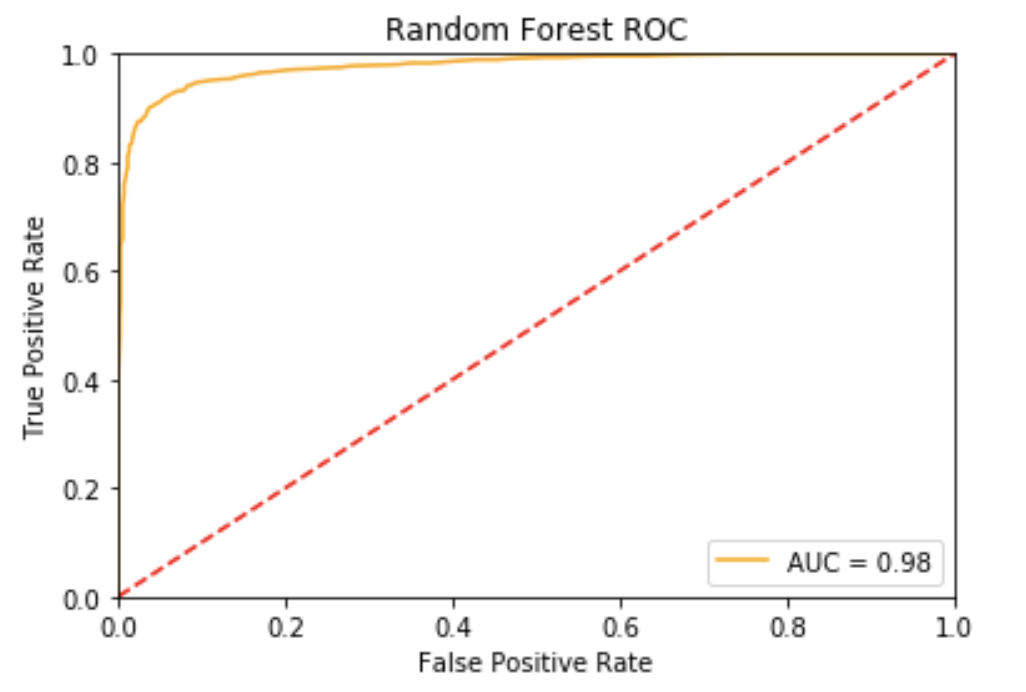
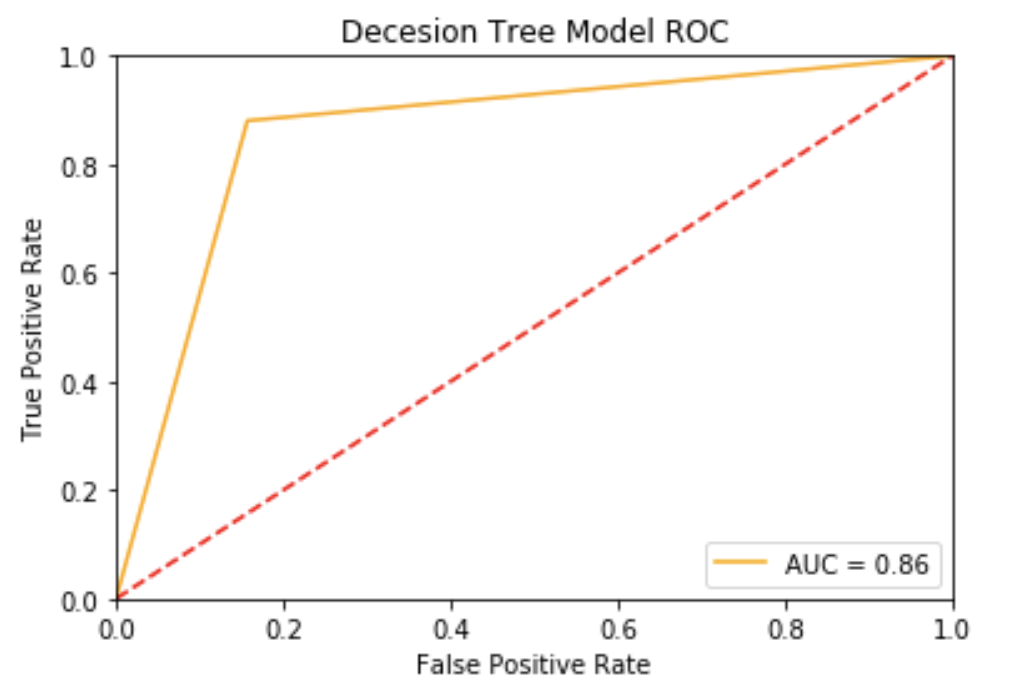
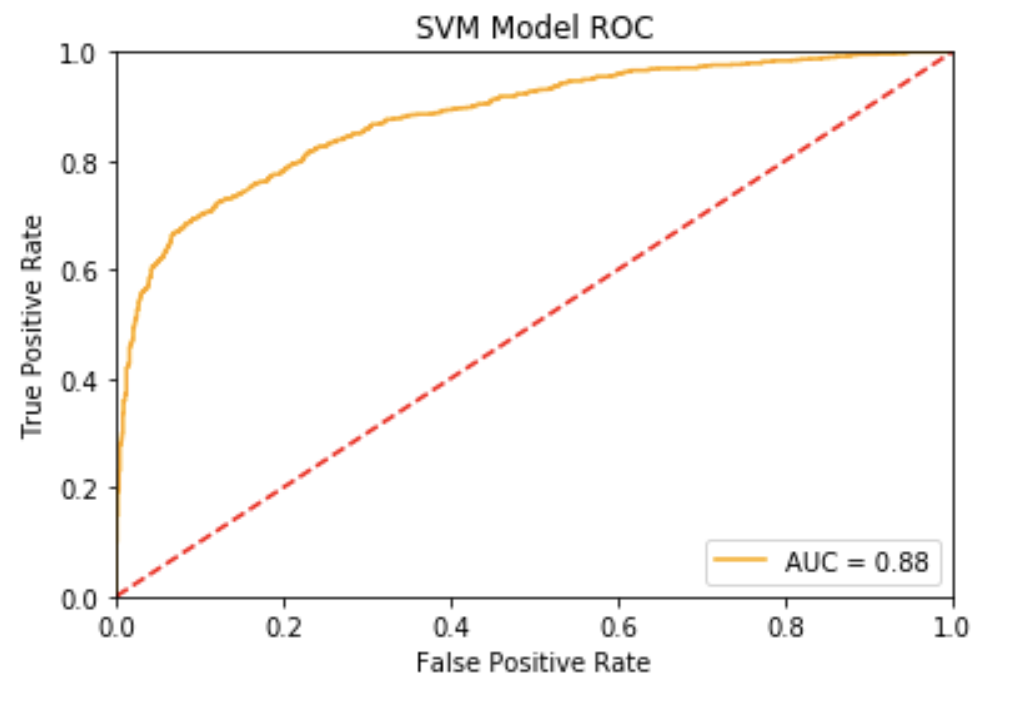
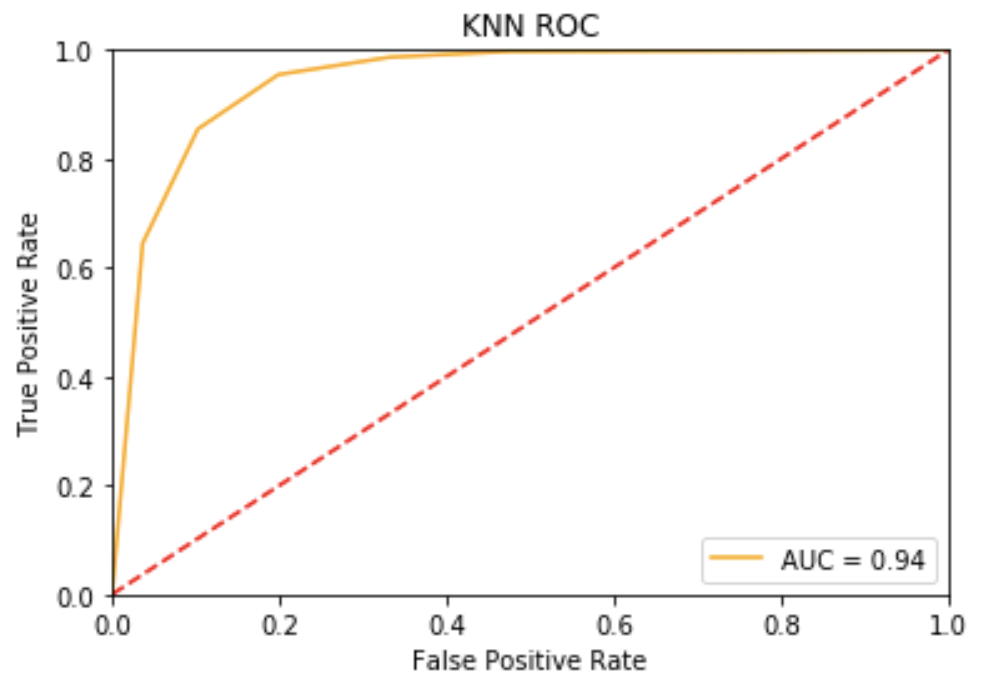
### **There are 5 models that we will investigate**

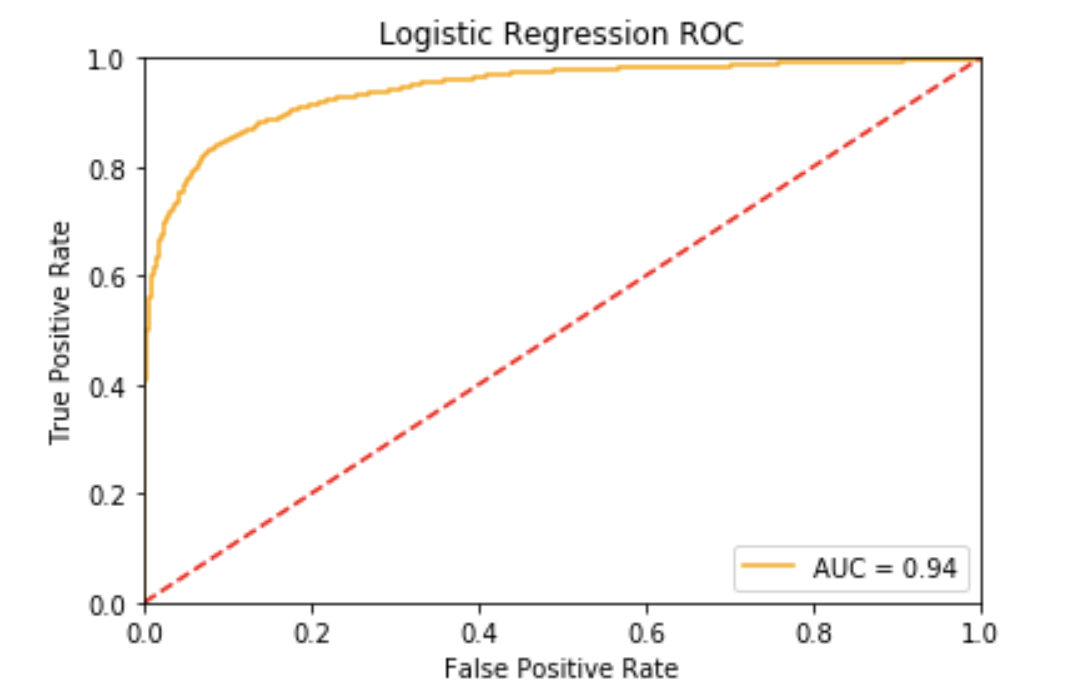
1. Logistic Regression
2. Decision Tree Classifier
3. Random Forest
4. KNeighborsClassifier
5. Support Vector Machine

### **1.1 With all available data**

We will use the default models just to get the idea how they work and then go deeper to each model. Also in the section below we will use all the available data.

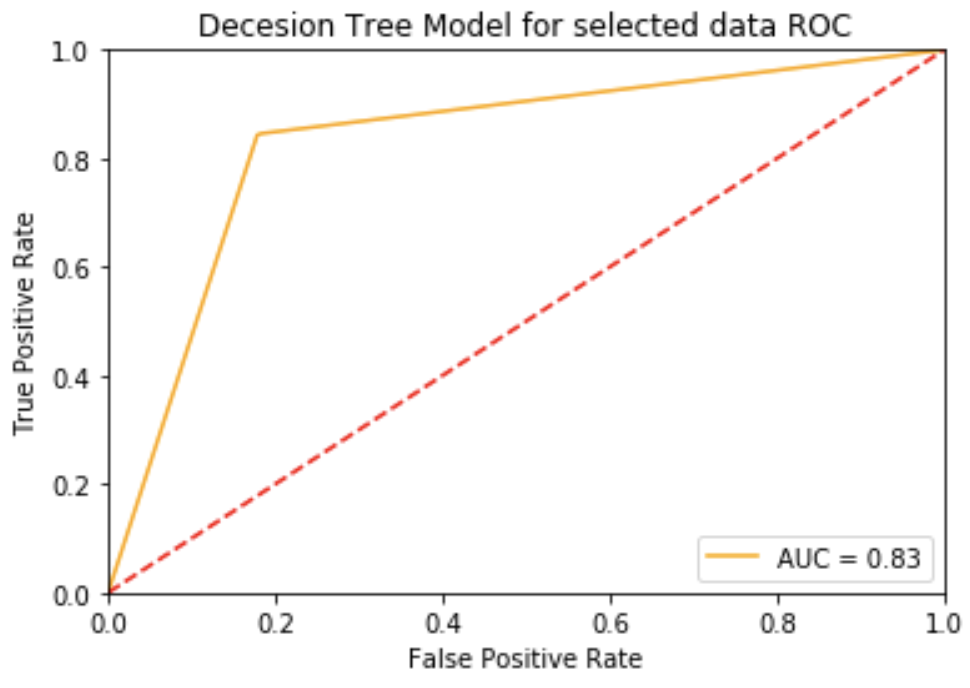
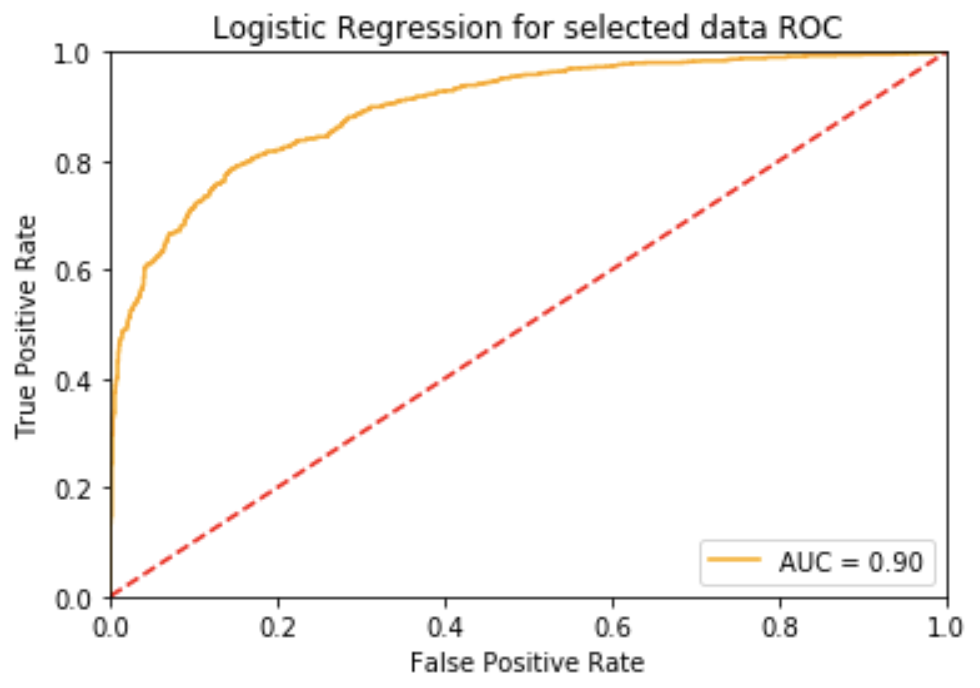
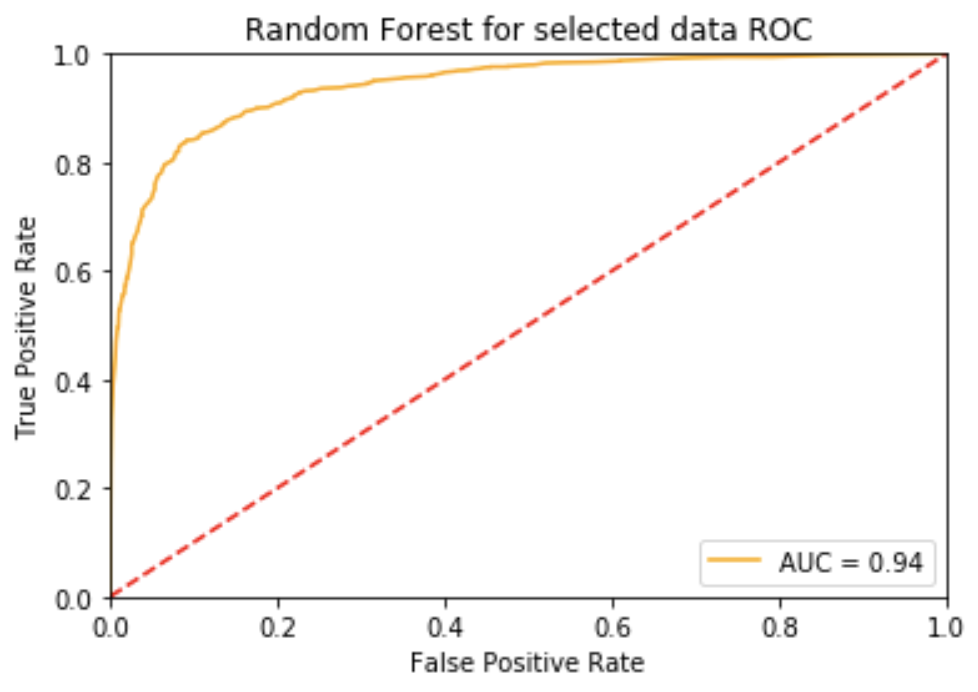
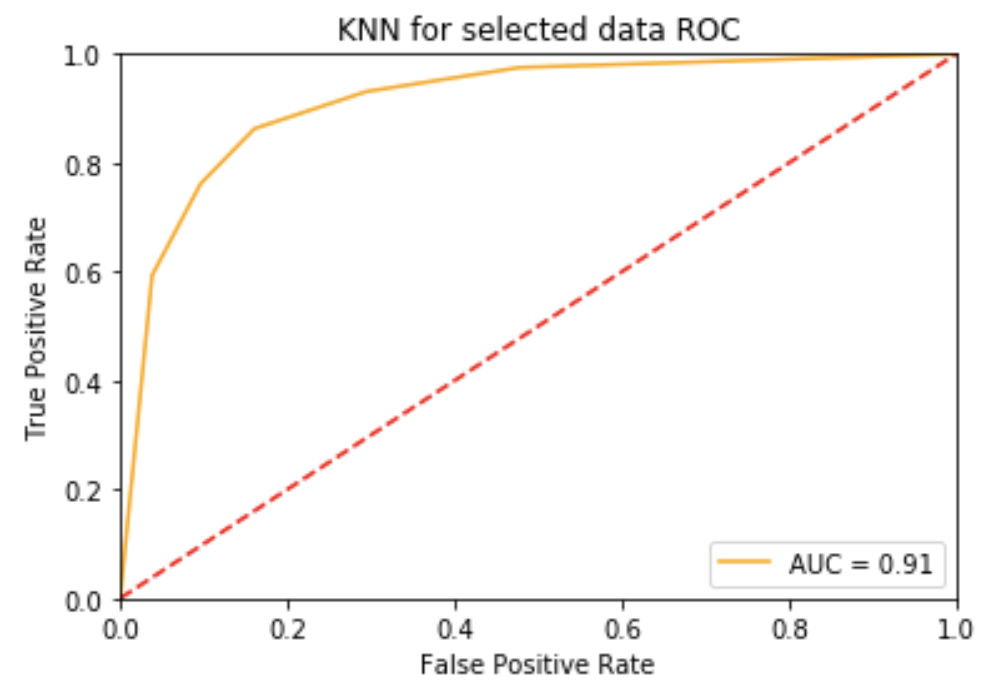
****

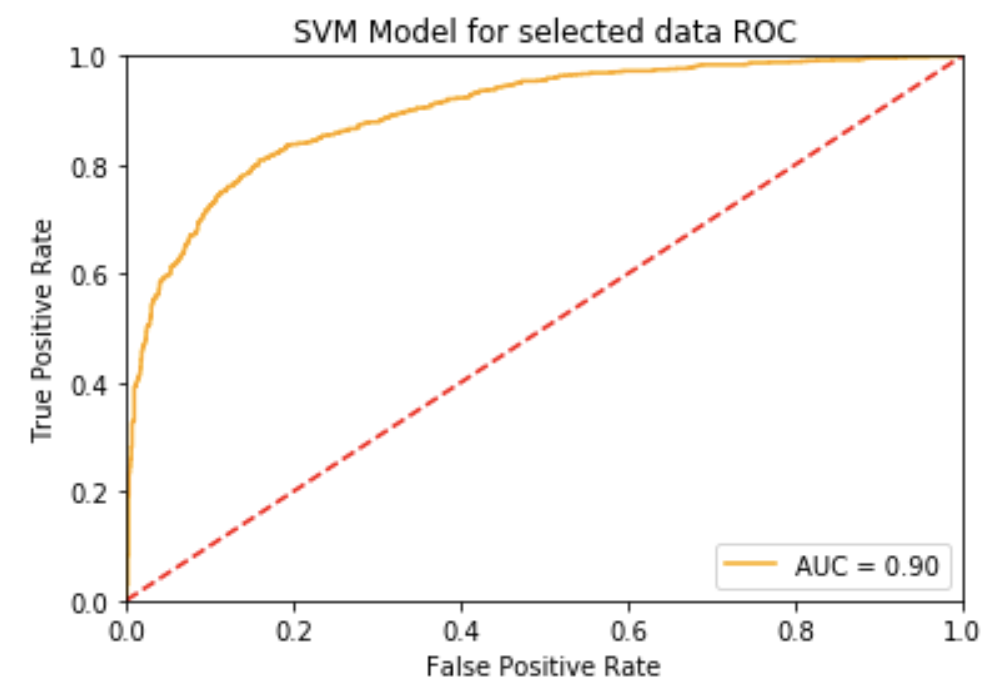




### **1.2 With selected features**

Now lets run the same classifiers on selected data with less features as below: ('Grade\_Grade II', 'Progesterone Status\_Positive', '6th Stage\_IIIC', 'Marital Status\_Married', 'Estrogen Status\_Positive')





### 

### 

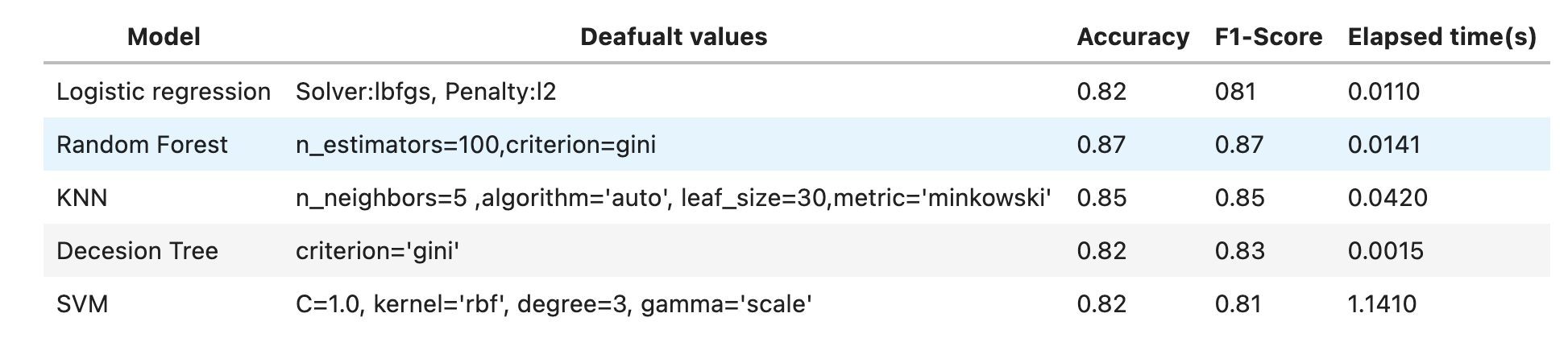
### 

### **Summary of the results**

Now lets see the result for all the available data

****

Now lets see the result for fewer selected data

****

Different classification models with default values, while using all the data or selected data gave us different results. As seen all models have very close Accuracy and F1-score when using all the data. Elapsed time is much higher in the SVM model, around 2.33 seconds.

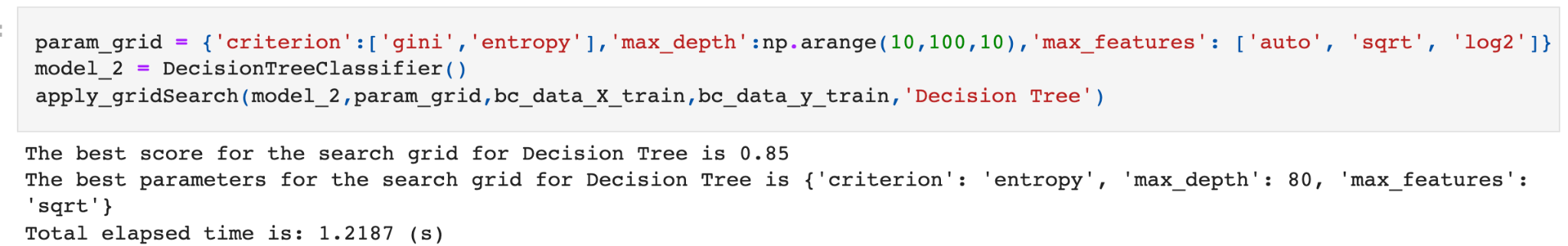
When decreasing the columns and used some specific features('Grade\_Grade II', 'Progesterone Status\_Positive', '6th Stage\_IIIC', 'Marital Status\_Married', 'Estrogen Status\_Positive') that have most of the data's property, the accuracy declined in all models almost by 10%. But as expected the elapsed time for running and predicting has decreased a lot.

In conclusion, as the data is not so huge and the elapsed time does not seem to be a problem for the amount of the data we have, and the accuracy is better with all the data, we will use all the data for deeper analysis.

### **Model Exploration with hyper parameter**

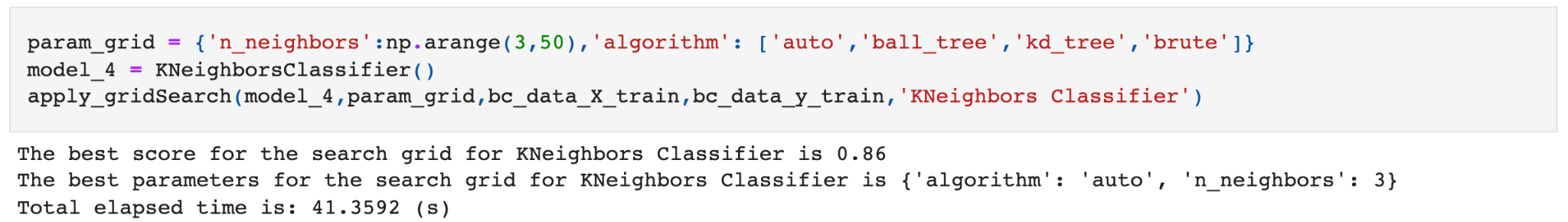
Model 1: Logistic Regression

### 

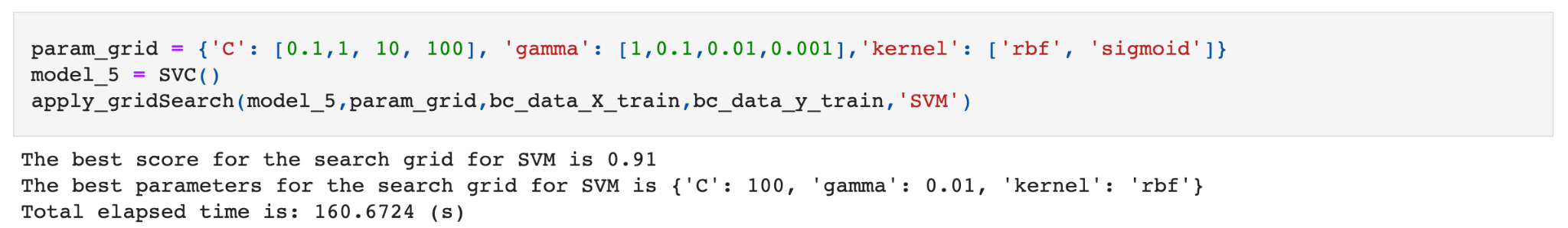
Model 2: Decision Tree Classifier

Model 3: Random Forest Classifier

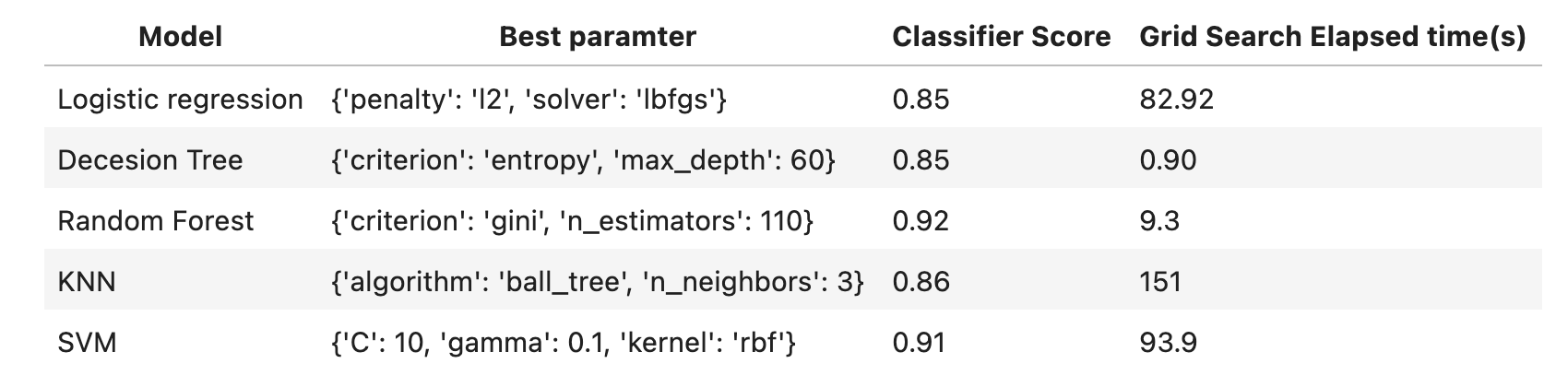


Model 4: KNeighbors Classifier

Model 5: Support Vector Machine

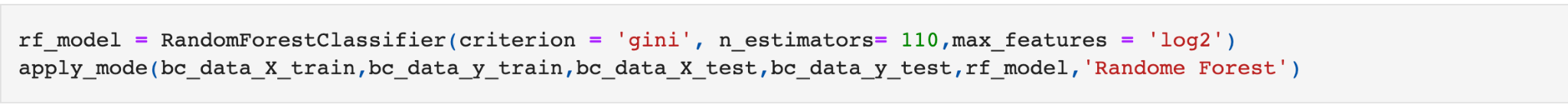


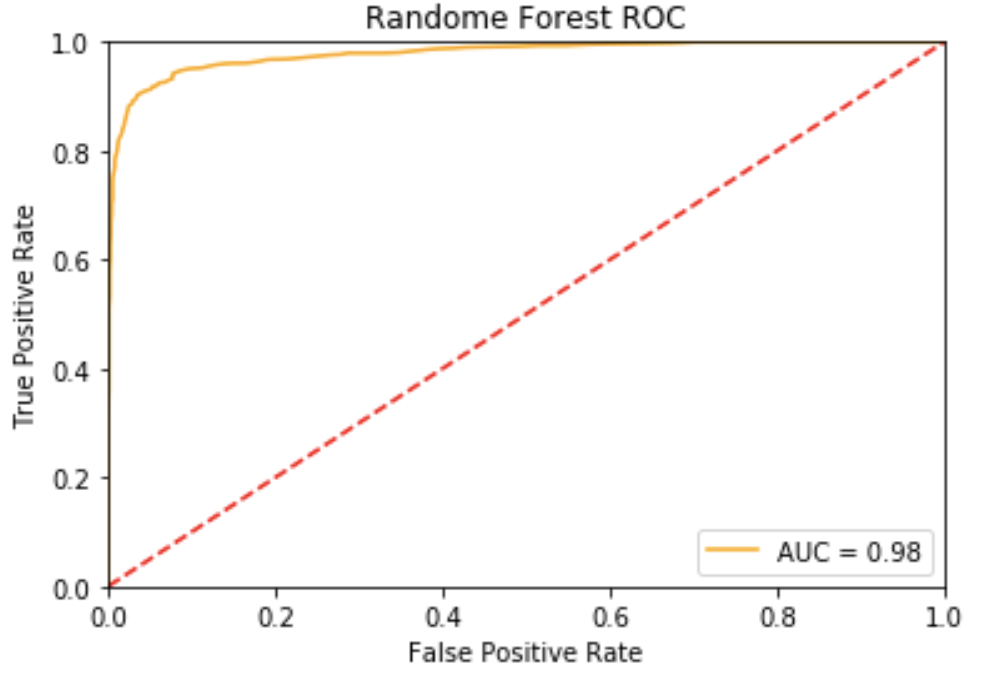
Let's summarize the result:



According to the different models' score result, Random Forest and SVM have the highest score equal to 92% AND 91%. Let's compare the performance of these models with the selected parameter.

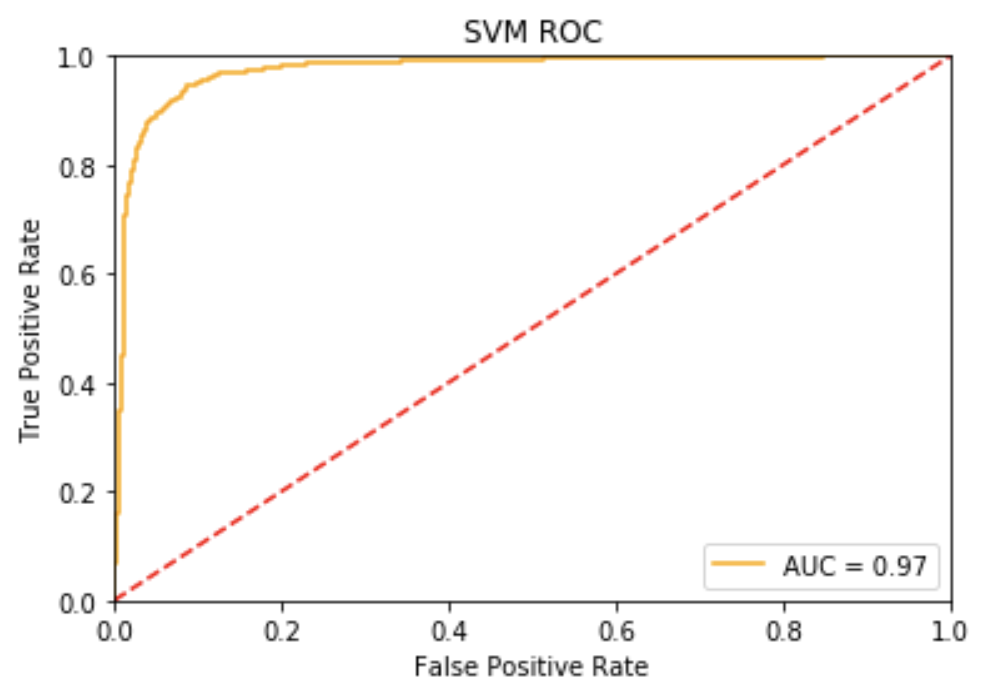
**RandomForestClassifier:**

****

****

**SupportVectorClassification.:**

****



### **Summary of the results**

### 

After running the models and comparing the result for predicting the test data we see slightly better AUC (98%) for RF compared to SVM which is 97%%. They both have an equal (93%) score. Also if we consider the performance of the models, Decision Tree took much much less time to fit (0.2877) compared to SVM fitting time (2.79), also much less time to predict the test data (0.03) compared to the SVM predicting time which is (0.11 s).

For our purpose, which is to predict the survival rate of cancer patients, we select Random Forest as our winning model.