```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.feature_selection import mutual_info_classif

from sklearn.metrics import accuracy_score, confusion_matrix, classification_report

from sklearn.linear_model import LogisticRegression
from sklearn. ensemble import RandomForestClassifier

import warnings
warnings.filterwarnings('ignore')
```

```
path = "/content/drive/MyDrive/Datasets/Kaggle/breast-cancer.csv"
df = pd.read_csv(path)
df.head()
```

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	poi
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	

5 rows × 32 columns

```
new_df = df
new_df['diagnosis'] = new_df['diagnosis'].replace({'M':1, 'B':0})
new_df.head()
```

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	poi
0	842302	1	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	
1	842517	1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	
2	84300903	1	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	
3	84348301	1	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	
4	84358402	1	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	

5 rows × 32 columns

```
df.columns
```

```
Index(['id', 'diagnosis', 'radius_mean', 'texture_mean', 'perimeter_mean',
       'area_mean', 'smoothness_mean', 'compactness_mean', 'concavity_mean',
       'concave points_mean', 'symmetry_mean', 'fractal_dimension_mean',
       'radius_se', 'texture_se', 'perimeter_se', 'area_se', 'smoothness_se',
       'compactness_se', 'concavity_se', 'concave points_se', 'symmetry_se',
       'fractal_dimension_se', 'radius_worst', 'texture_worst',
       'perimeter_worst', 'area_worst', 'smoothness_worst',
       'compactness_worst', 'concavity_worst', 'concave points_worst',
       'symmetry_worst', 'fractal_dimension_worst'],
      dtype='object')
```

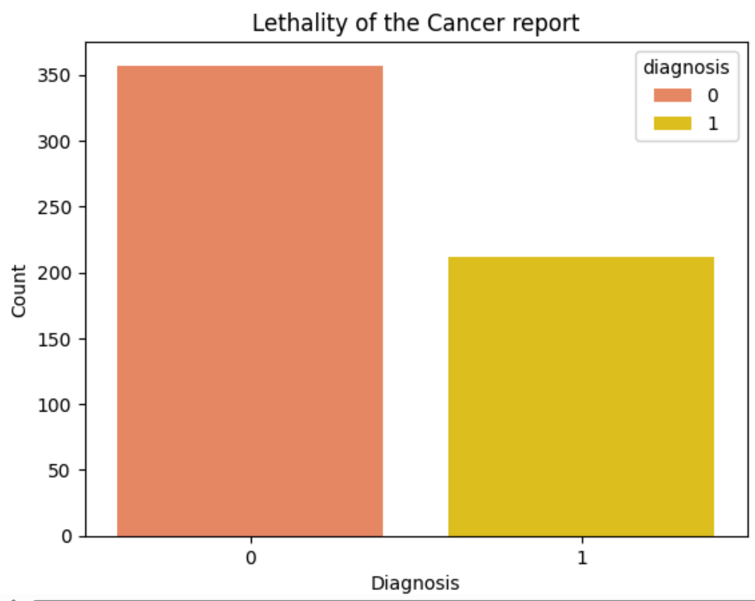
```
#Checking for null values
null_values = df.isnull().sum()
print(null_values)
```

```
id                         0
diagnosis                  0
radius_mean                0
texture_mean               0
perimeter_mean             0
area_mean                  0
smoothness_mean            0
compactness_mean           0
concavity_mean             0
concave points_mean        0
```

```
symmetry_mean              0
fractal_dimension_mean     0
radius_se                  0
texture_se                 0
perimeter_se               0
area_se                    0
smoothness_se              0
compactness_se             0
concavity_se               0
concave points_se          0
symmetry_se                0
fractal_dimension_se       0
radius_worst               0
texture_worst              0
perimeter_worst            0
area_worst                 0
smoothness_worst           0
compactness_worst          0
concavity_worst            0
concave points_worst       0
symmetry_worst             0
fractal_dimension_worst    0
dtype: int64
```

The visualization below indicates that there are more number of **Benign tumors as compared to Malignant tumors** in the population considered for biopsy.

```
sns.countplot(data = df, x = 'diagnosis', hue = 'diagnosis', palette = ['coral', 'gold'])
plt.xlabel("Diagnosis")
plt.ylabel("Count")
plt.title("Lethality of the Cancer report")
plt.show()
```



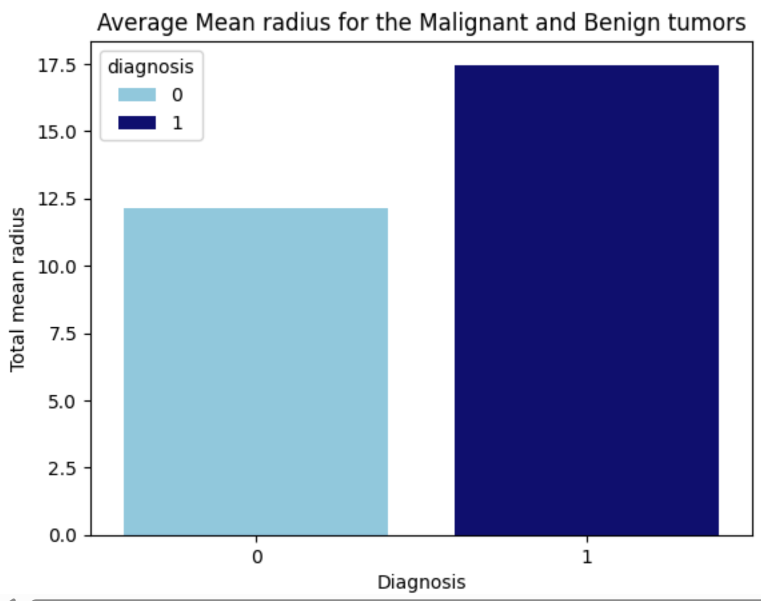
Mean radius analysis between the two groups which suggests that the **average radius of Malignant tumors is higher as compared to Benign tumors**. A higher radius mean value can indicate towards the **cells having larger nuclei** which is a key characteristic in diagnosing cancer reflecting an **increased likelihood of malignancy**

```
total_mean_radius = df.groupby('diagnosis')['radius_mean'].mean()
total_mean_radius.reset_index()
```

	diagnosis	radius_mean
0	0	12.146524
1	1	17.462830

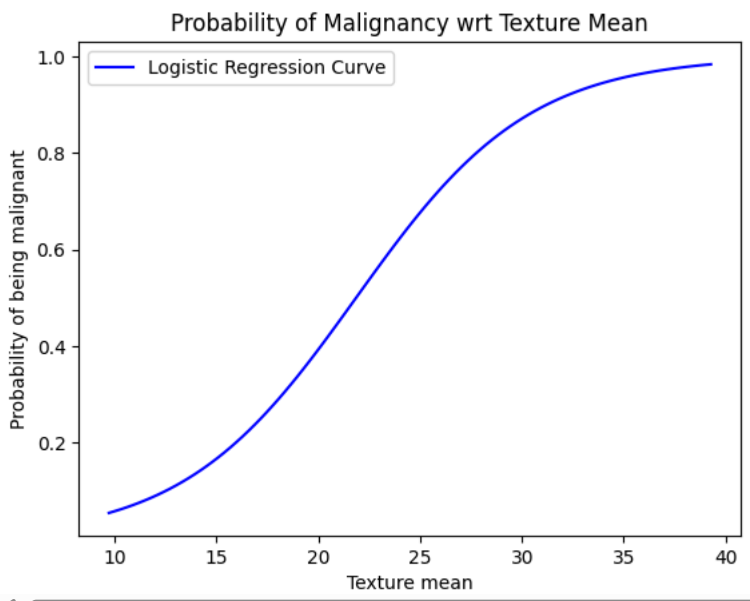
A graphical representation of the above performed **Mean radius analysis**

```
total_mean_radius = df.groupby('diagnosis')['radius_mean'].mean().reset_index()

sns.barplot(data = total_mean_radius, x = 'diagnosis', y = 'radius_mean', hue = 'diagnosis', palette = ['skyblue', 'navy'])
plt.xlabel("Diagnosis")
plt.ylabel("Total mean radius")
plt.title("Average Mean radius for the Malignant and Benign tumors")
plt.show()
```



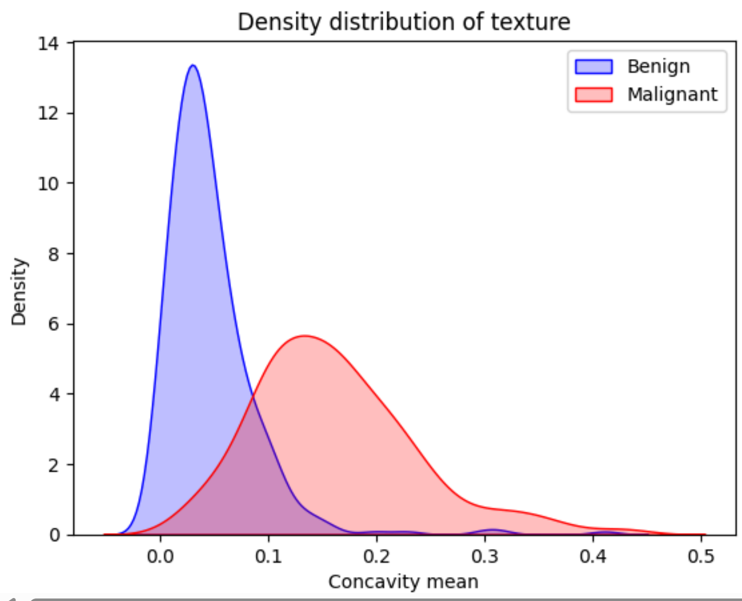
Through the **Logistic Regression Curve** below, it can be interpreted that the **likelihood of Malignancy increases with increase in Texture Mean**

```
x = df[['texture_mean']]
y = df['diagnosis']

model = LogisticRegression()
model.fit(x, y)

x_range = np.linspace(start = df['texture_mean'].min(), stop = df['texture_mean'].max(), num = 100).reshape(-1, 1)
y_prob = model.predict_proba(x_range)[ : , 1] #Extracting only the probability of being malignant (class 1)

plt.plot(x_range, y_prob, color = 'b', label = 'Logistic Regression Curve')
plt.xlabel("Texture mean")
plt.ylabel("Probability of being malignant")
plt.title("Probability of Malignancy wrt Texture Mean")
plt.legend()
plt.show()
```



The **blue curve represents the density distribution of concavity mean for Benign tumors with the peak of the curve being much higher and closer to 0.0** than the **red curve for Malignant tumors** indcating that **most Benign tumors have much lower concavity mean**. This means that the **Benign tumors have less concave features** on their cell surfaces

```
sns.kdeplot(data = new_df[new_df['diagnosis']==0], x = 'concavity_mean', color = 'blue', fill = True, label = "Benign")
sns.kdeplot(data = new_df[new_df['diagnosis']==1], x = 'concavity_mean', color = 'red', fill = True, label = "Malignant")
plt.xlabel("Concavity mean")
plt.ylabel("Density")
plt.title("Density distribution of texture")
plt.legend()
plt.show()
```



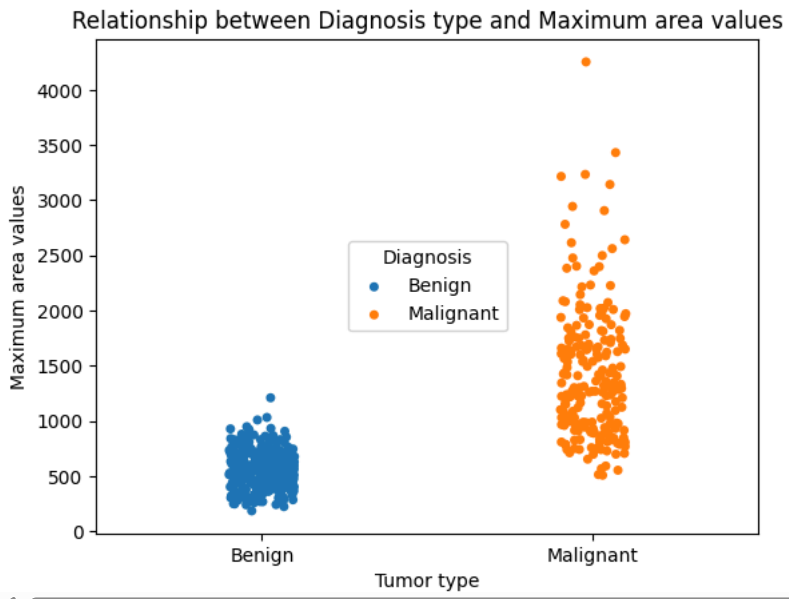
Mean maximum-area analysis between the two groups to compare the maximum area records of both the type of tumors

```
ar_worst_plot = df.groupby('diagnosis')["area_worst"].mean().reset_index()
ar_worst_plot
```

	diagnosis	area_worst
0	0	558.899440
1	1	1422.286321

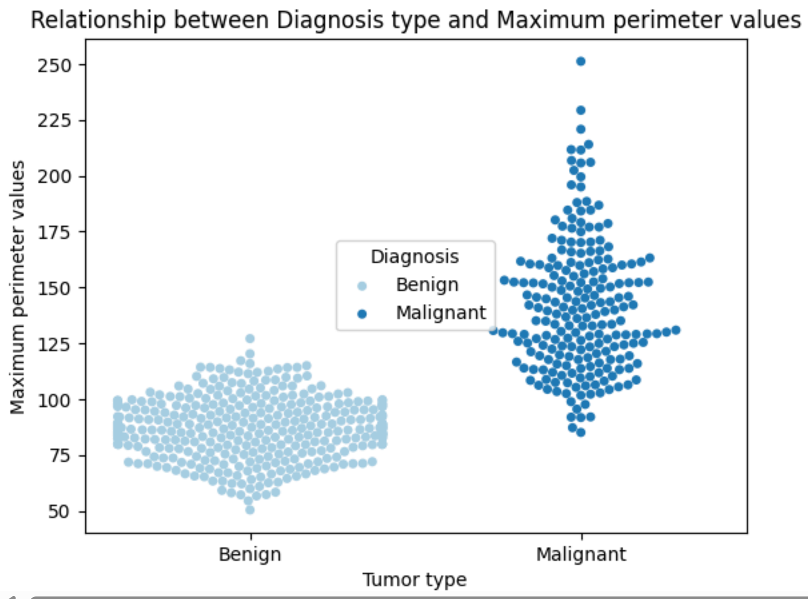
The **Stripplot** below proves that **maximum area values for Malignant type is higher** as compared to Benign type as **Malignant cells in general have larger nuclei** since they tend to **grow uncontrollably resulting in abnormal cell size**

```
sns.stripplot(data = new_df, x = "diagnosis", y = "area_worst", hue = "diagnosis")
plt.xlabel("Tumor type")
plt.ylabel("Maximum area values")
plt.title("Relationship between Diagnosis type and Maximum area values")
plt.legend(title = "Diagnosis", labels = ["Benign", "Malignant"], loc = "center")
plt.xticks(ticks = [0, 1], labels = ["Benign", "Malignant"])
plt.show()
```



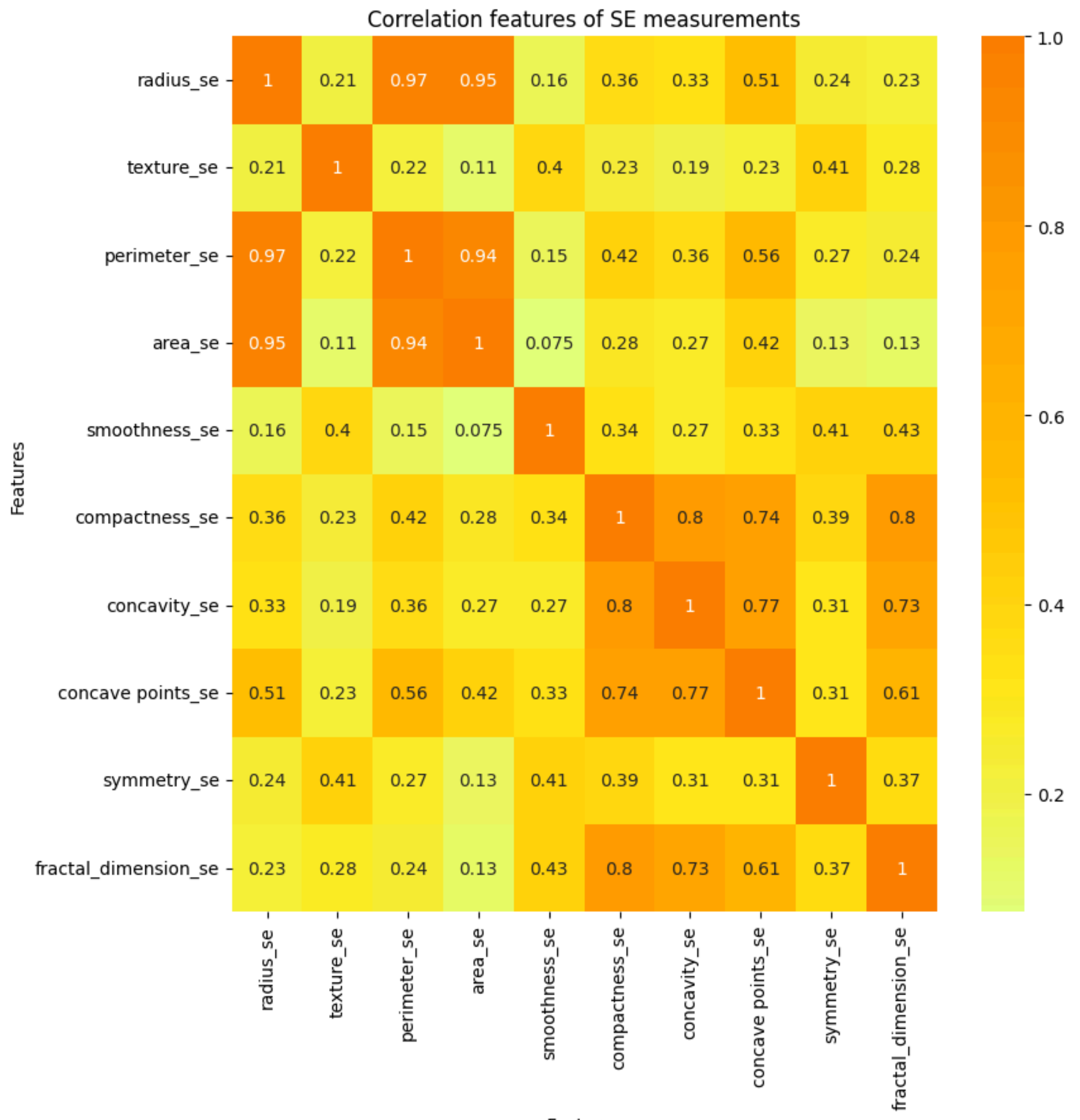
The graphical representation below proves that **Malignant tumors** record to have **higher maximum perimeter values**

```
sns.swarmplot(data = new_df, x = "diagnosis", y = "perimeter_worst", hue = "diagnosis", palette = "Paired")
plt.xlabel("Tumor type")
plt.ylabel("Maximum perimeter values")
plt.title("Relationship between Diagnosis type and Maximum perimeter values")
plt.legend(title = "Diagnosis", labels = ["Benign", "Malignant"], loc = "center")
plt.xticks(ticks = [0, 1], labels = ["Benign", "Malignant"])
plt.show()
```



Using **Pearson's Correlation** to observe the correlated features

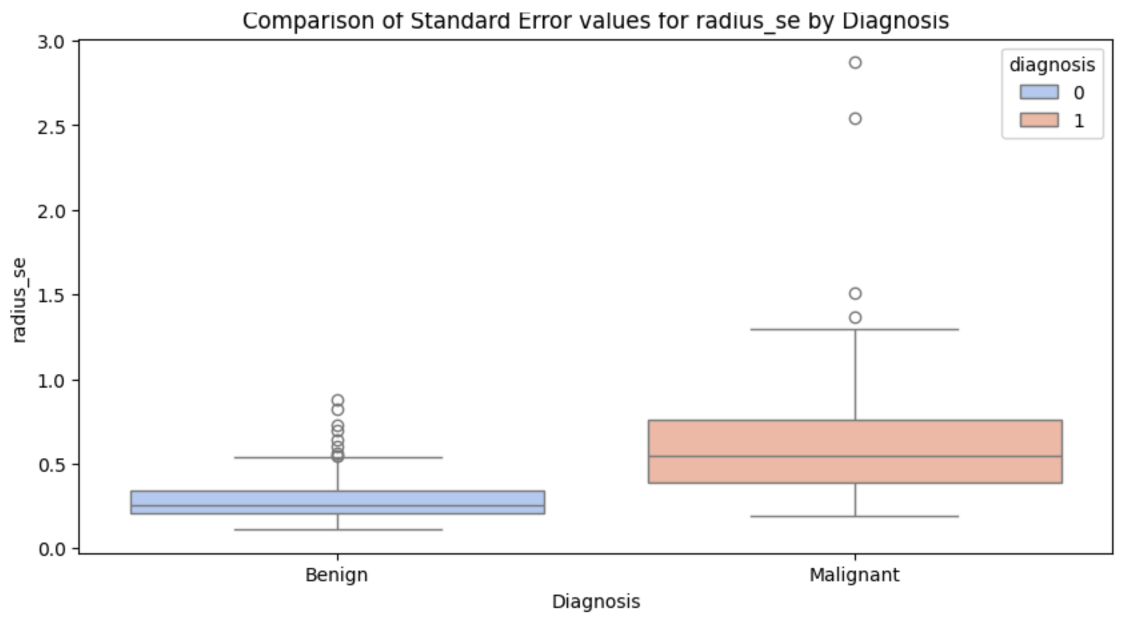
```
se_cols = new_df.filter(items = ['radius_se', 'texture_se', 'perimeter_se', 'area_se', 'smoothness_se',
       'compactness_se', 'concavity_se', 'concave points_se', 'symmetry_se',
       'fractal_dimension_se'])
cor = se_cols.corr()
plt.figure(figsize = (9, 9))
sns.heatmap(cor, annot = True, cmap = 'Wistia', fmt = ".2g")
plt.xlabel("Features")
plt.ylabel("Features")
plt.title("Correlation features of SE measurements")
plt.show()
```

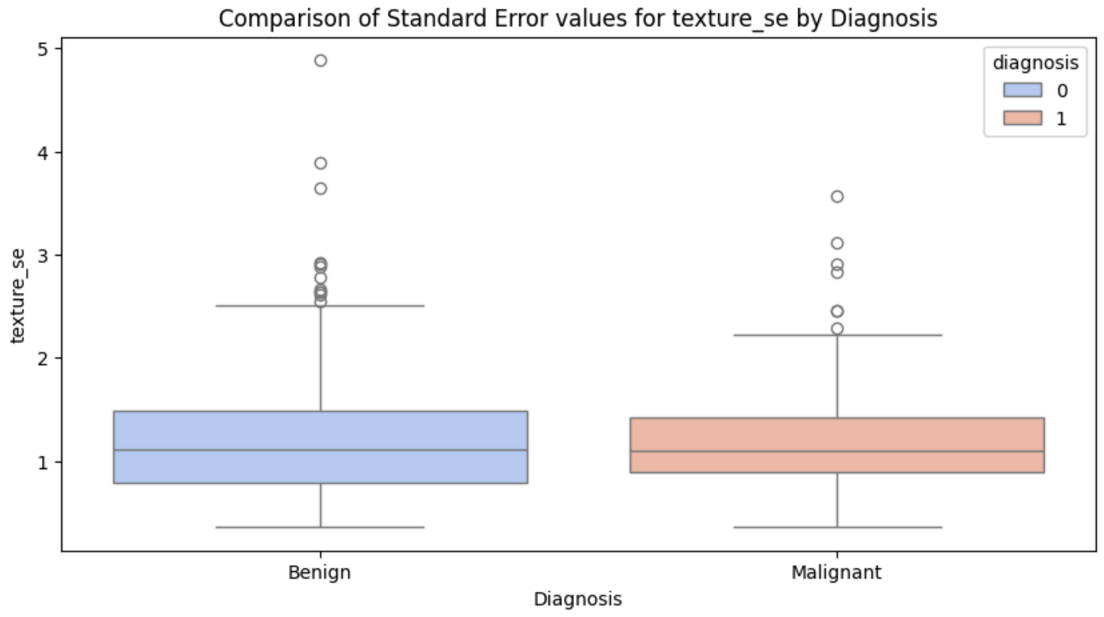


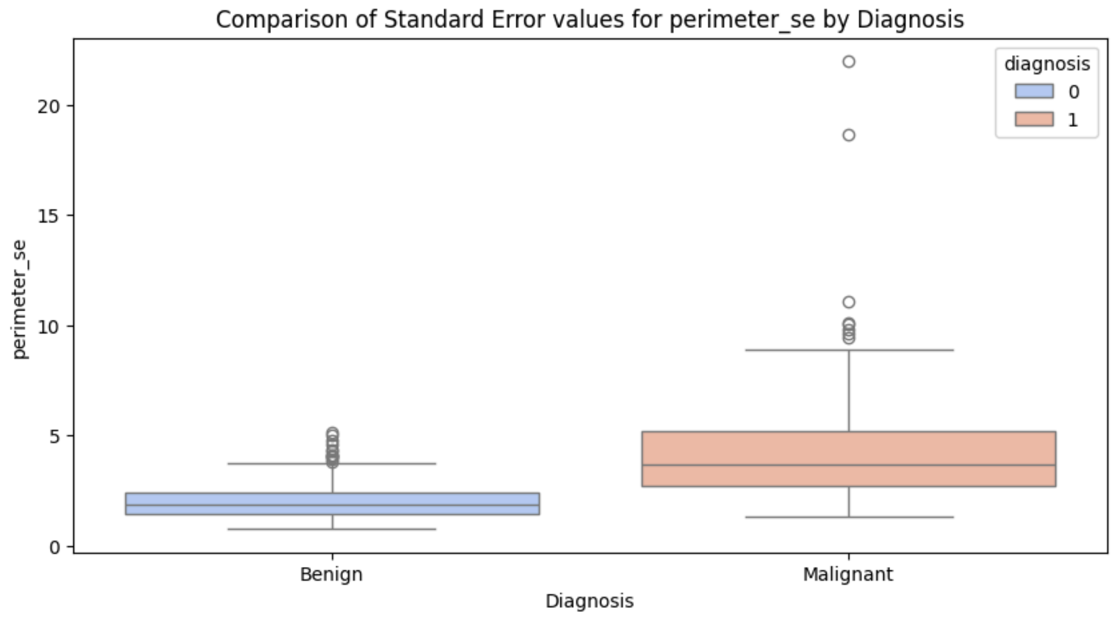
	radius_se	texture_se	perimeter_se	area_se	smoothness_se	compactness_se	concavity_se	concave points_se	symmetry_se	fractal_dimension_se
radius_se	1	0.21	0.97	0.95	0.16	0.36	0.33	0.51	0.24	0.23
texture_se	0.21	1	0.22	0.11	0.4	0.23	0.19	0.23	0.41	0.28
perimeter_se	0.97	0.22	1	0.94	0.15	0.42	0.36	0.56	0.27	0.24
area_se	0.95	0.11	0.94	1	0.075	0.28	0.27	0.42	0.13	0.13
smoothness_se	0.16	0.4	0.15	0.075	1	0.34	0.27	0.33	0.41	0.43
compactness_se	0.36	0.23	0.42	0.28	0.34	1	0.8	0.74	0.39	0.8
concavity_se	0.33	0.19	0.36	0.27	0.27	0.8	1	0.77	0.31	0.73
concave points_se	0.51	0.23	0.56	0.42	0.33	0.74	0.77	1	0.31	0.61
symmetry_se	0.24	0.41	0.27	0.13	0.41	0.39	0.31	0.31	1	0.37
fractal_dimension_se	0.23	0.28	0.24	0.13	0.43	0.8	0.73	0.61	0.37	1

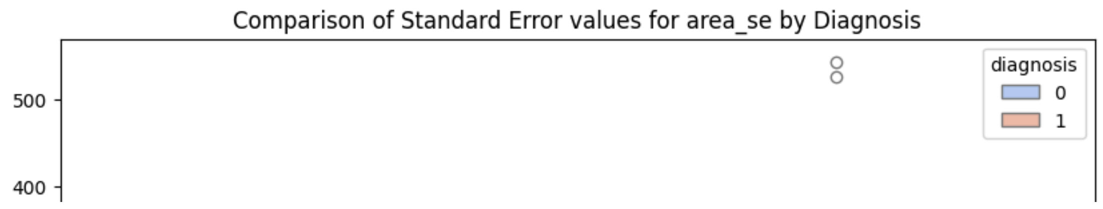
The plot below helps us understand if malignant and benign tumors have different **Standard Error** values for all features and if there are outliers

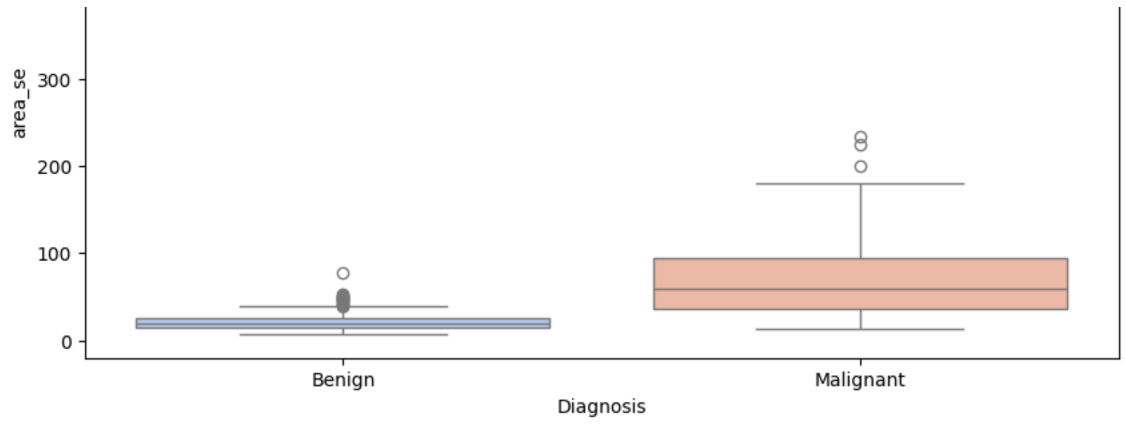
```
se_cols2 = new_df.filter(items = ['diagnosis', 'radius_se', 'texture_se', 'perimeter_se', 'area_se', 'smoothness_se',
       'compactness_se', 'concavity_se', 'concave points_se', 'symmetry_se','fractal_dimension_se'])
for col in se_cols2.columns:
  if col == 'diagnosis':
    continue
  else:
    plt.figure(figsize = (10, 5))
    sns.boxplot(data = se_cols2, x = 'diagnosis', y = col, hue = 'diagnosis', palette = "coolwarm")
    plt.title(f"Comparison of Standard Error values for {col} by Diagnosis")
    plt.xlabel("Diagnosis")
    plt.xticks(ticks = [0, 1], labels = ['Benign', 'Malignant'])
    plt.ylabel(f"{col}")
    plt.show()
```


Comparison of Standard Error values for radius_se by Diagnosis
radius_se
Diagnosis
Benign
Malignant
diagnosis
0
1

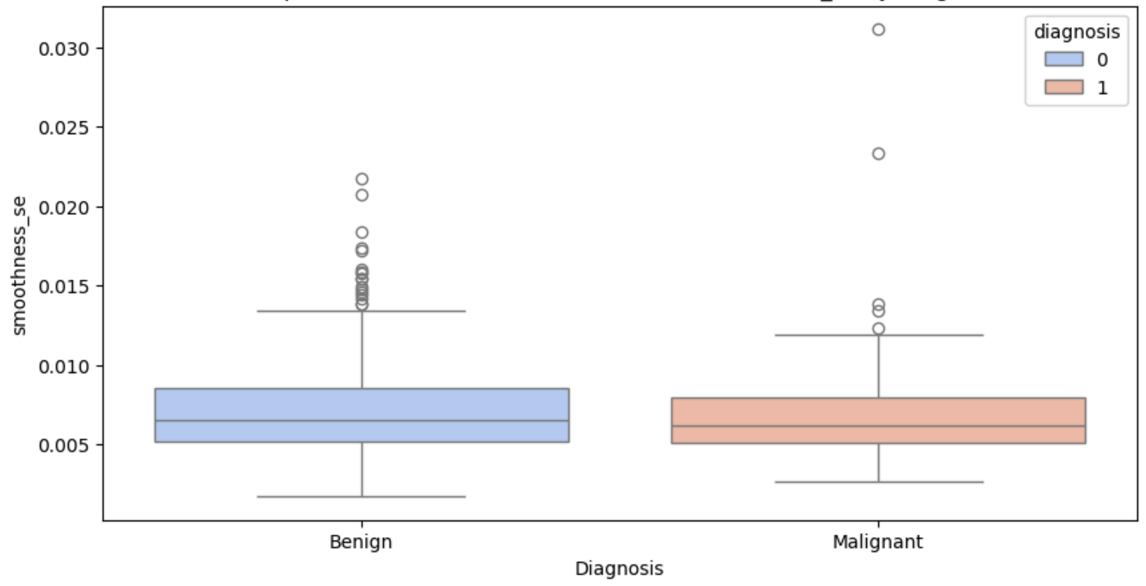

Comparison of Standard Error values for texture_se by Diagnosis
texture_se
Diagnosis
Benign
Malignant
diagnosis
0
1


Comparison of Standard Error values for perimeter_se by Diagnosis
perimeter_se
Diagnosis
Benign
Malignant
diagnosis
0
1

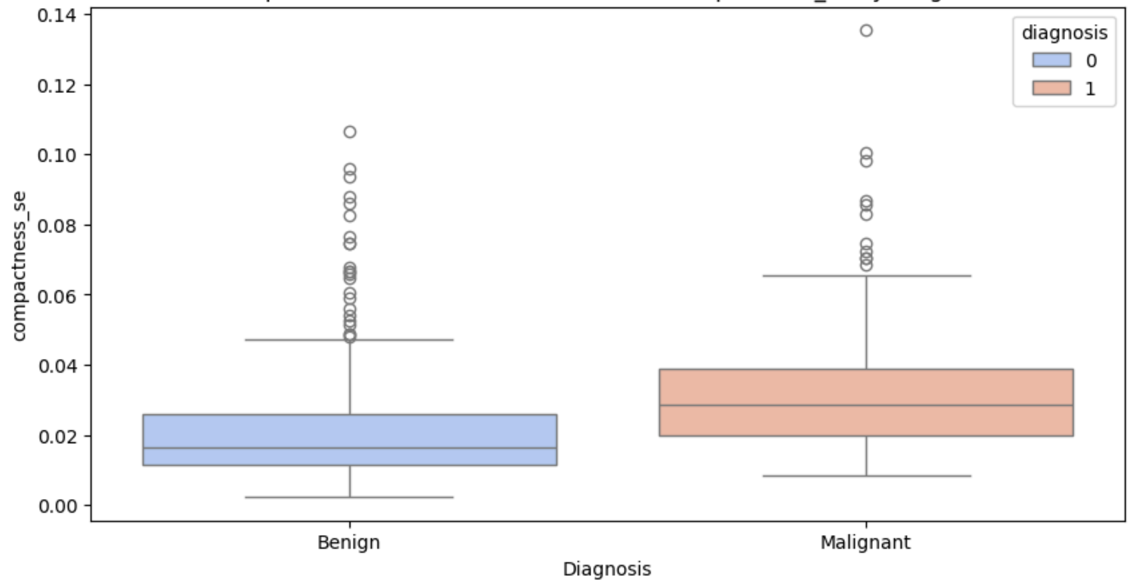

Comparison of Standard Error values for area_se by Diagnosis
diagnosis
0
1



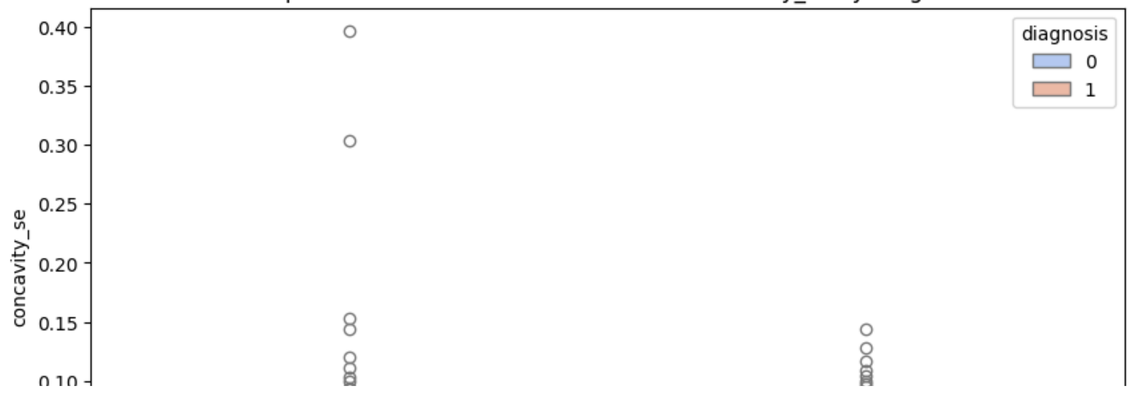
Comparison of Standard Error values for smoothness_se by Diagnosis

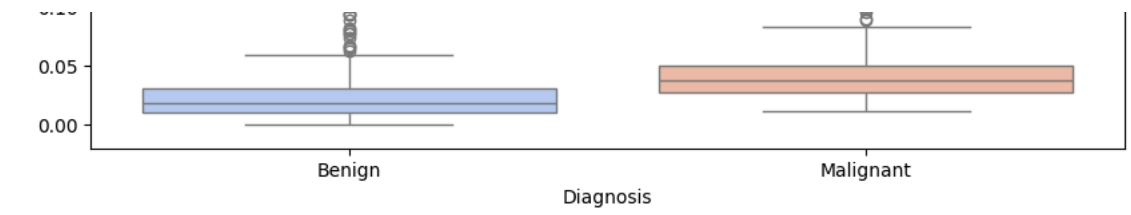


Comparison of Standard Error values for compactness_se by Diagnosis

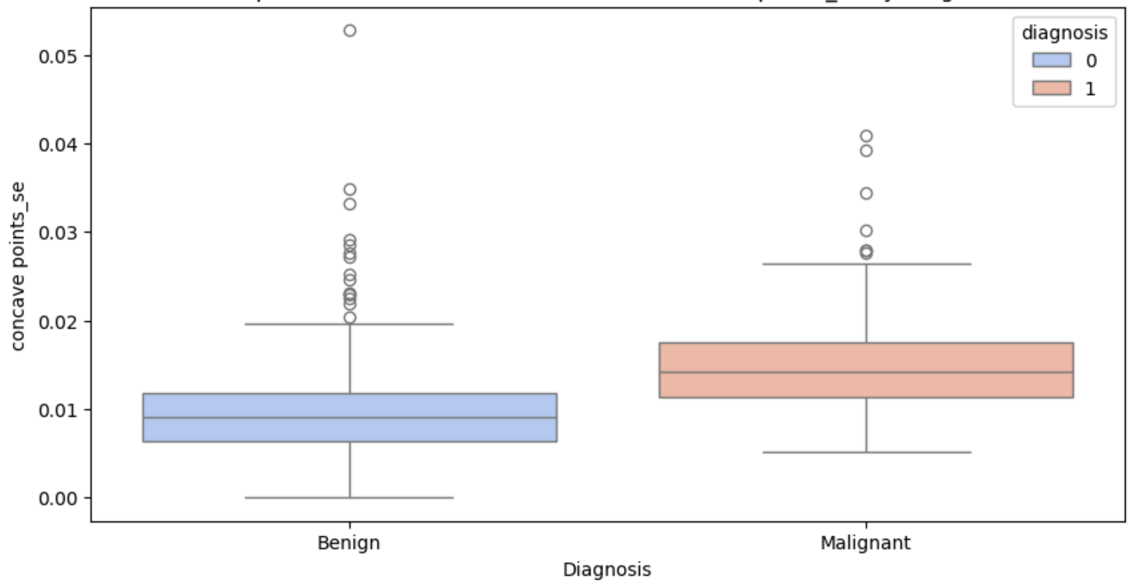


Comparison of Standard Error values for concavity_se by Diagnosis

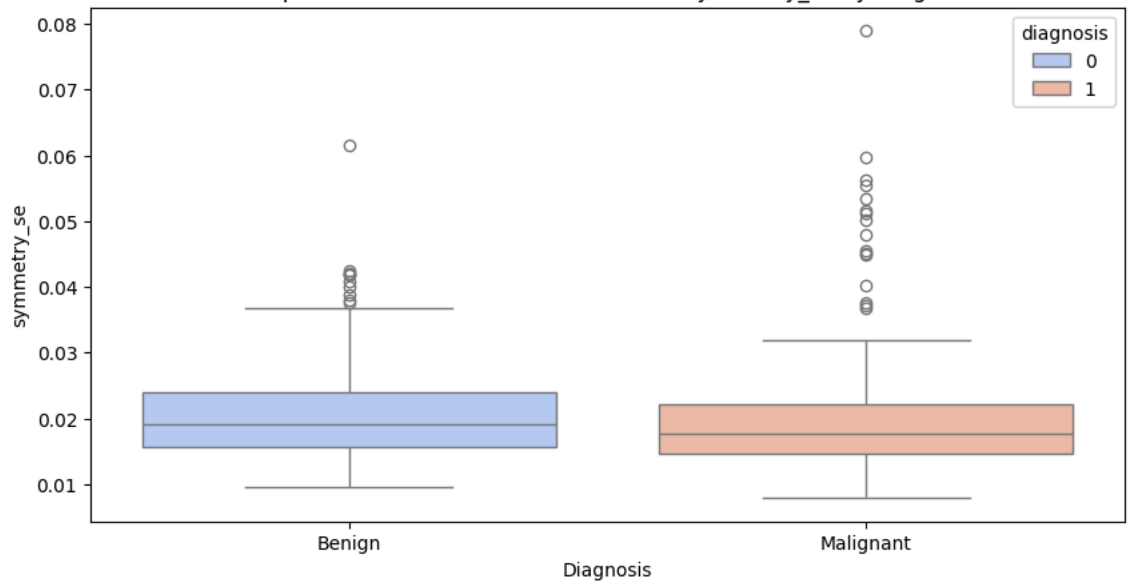




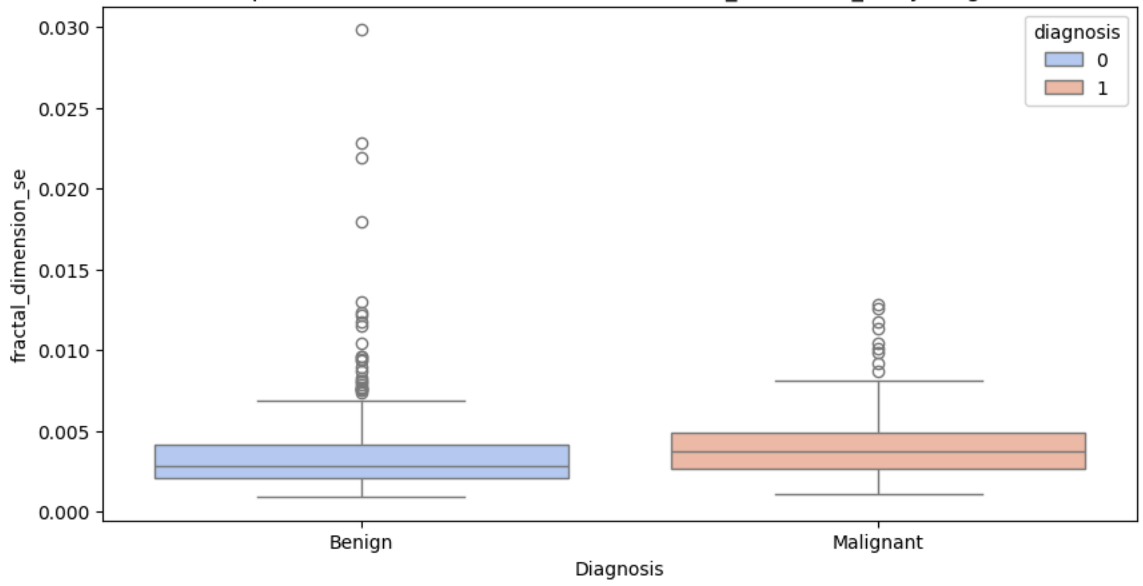
Comparison of Standard Error values for concave points_se by Diagnosis



Comparison of Standard Error values for symmetry_se by Diagnosis



Comparison of Standard Error values for fractal_dimension_se by Diagnosis



Train Test split

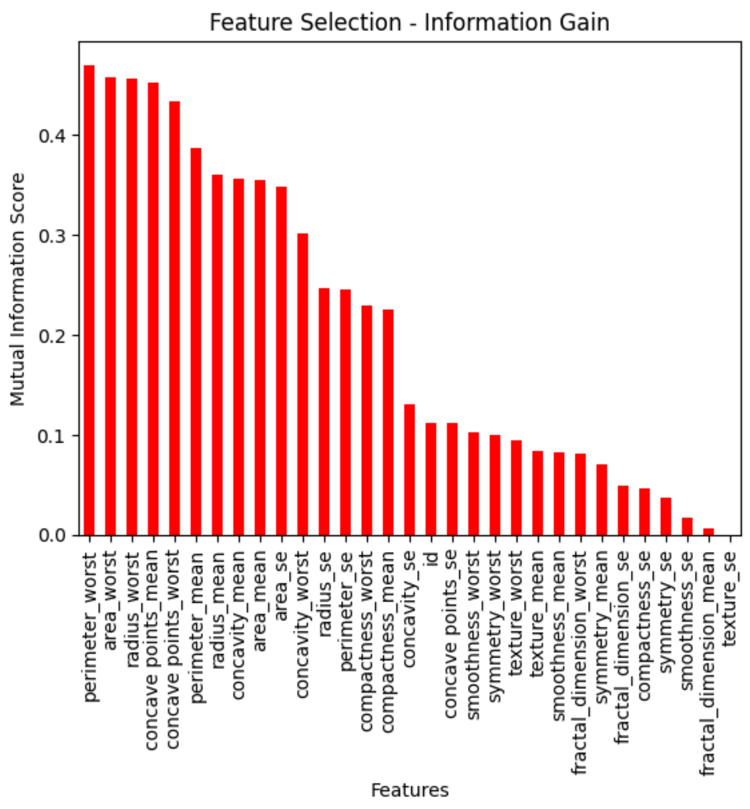
```
df2 = new_df.drop('diagnosis', axis = 1)
target = new_df[['diagnosis']]
X_train, X_test, y_train, y_test = train_test_split(df2, target, test_size = 0.2, random_state = 30)
```

Feature Selection using Information Gain technique

```
mutual_info_score = mutual_info_classif(X_train, y_train)
mutual_info_score
```

```
array([0.11210944, 0.36083745, 0.08382783, 0.3875275 , 0.35575182,
       0.08193858, 0.22596127, 0.35631244, 0.4528174 , 0.0698089 ,
       0.00651145, 0.24681156, 0.        , 0.24505583, 0.34844999,
       0.01719469, 0.04640519, 0.13010502, 0.11201349, 0.03632867,
       0.04856441, 0.45615581, 0.09470747, 0.47063929, 0.45857913,
       0.10305207, 0.22991756, 0.30112416, 0.43351105, 0.09915416,
       0.08122484])
```

```
mutual_info_score = pd.Series(mutual_info_score, index = X_train.columns, name = "Mutual Information Score assigned")
mutual_info_score.sort_values(ascending = False).plot(kind = "bar", color = 'r')
plt.xlabel("Features")
plt.ylabel("Mutual Information Score")
plt.title("Feature Selection - Information Gain")
plt.figure(figsize = (10, 10))
plt.show()
```



```
<Figure size 1000x1000 with 0 Axes>
```

Random Forest

```
max_acc = 0

for i in range(100):
  model = RandomForestClassifier(random_state = i)
  model.fit(X_train, y_train)
  y_pred_rf = model.predict(X_test)
  current_acc = round(accuracy_score(y_test, y_pred_rf)*100, 2)
  if current_acc > max_acc:
    max_acc = current_acc
    best_rs = i

model = RandomForestClassifier(random_state = best_rs)
model.fit(X_train, y_train)
y_pred_rf = model.predict(X_test)
```