In [1]:
```
### Project Topic

# The purpose of this project is to classify breast cancer tumors as (benin or malignant).
# This project will be using a simple multi-linear regression model.
# Data source would come from the University of California, Irvine.
# This project would help doctors and physicians who have trouble classifying mTBI especially at initial tests.
```

In [168]:
```
### Data

# The data consist of an ID number, Diagnosis (M=malignant, B=Benign), and 30 features (tumor radius, texture, area, smoothness...)
# 569 subjects would be used for this exercise.

# Data was obtained from UCI's machine learning repository
# https://archive.ics.uci.edu/dataset/17/breast+cancer+wisconsin+diagnostic

import scipy as sp
import scipy.stats as stats
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import copy
# Set color map to have light blue background
sns.set()
import statsmodels.formula.api as smf
import statsmodels.api as sm
%matplotlib inline
from sklearn.model_selection import train_test_split
from sklearn.metrics import roc_curve, auc, f1_score, roc_auc_score


# Replace file_path with location of data.csv
file_path = r'C:\Users\Eddie\Desktop\Engineering\wdbc_wheaders.csv'
df = pd.read_csv(file_path)
```

In [169]:
```
### Data Cleaning - Display DF sample

# Data is initially .data file. File is processed externaly to add headers and convert to a csv file.
# Realized data would be 570 x 32 matrix. A total of 30 features and 569 samples.
# Data is clean and does not require much processing. None of the data has NAN and data is label in an appropriate category.
# Only the diagnosis is cleaned to convert Benign to 0 and Malignant to 1.

#              ID    DIAGNOSIS    RADIUS1    TEXTURE1    PERIMETER1 ... FRACTAL_DIMENSION3
# SUBJECT0
# SUBJECT1
# SUBJECT2
# SUBJECT3
#   ...
# SUBJECT568


print(df.head()) # Display df
```

```
         ID Diagnosis  radius1  texture1  perimeter1   area1  smoothness1  \
0    842302         M    17.99     10.38      122.80  1001.0      0.11840   
1    842517         M    20.57     17.77      132.90  1326.0      0.08474   
2  84300903         M    19.69     21.25      130.00  1203.0      0.10960   
3  84348301         M    11.42     20.38       77.58   386.1      0.14250   
4  84358402         M    20.29     14.34      135.10  1297.0      0.10030   

   compactness1  concavity1  concave_points1         ...           radius3  \
0       0.27760      0.3001          0.14710         ...             25.38   
1       0.07864      0.0869          0.07017         ...             24.99   
2       0.15990      0.1974          0.12790         ...             23.57   
3       0.28390      0.2414          0.10520         ...             14.91   
4       0.13280      0.1980          0.10430         ...             22.54   

   texture3  perimeter3   area3  smoothness3  compactness3  concavity3  \
0     17.33      184.60  2019.0       0.1622        0.6656      0.7119   
1     23.41      158.80  1956.0       0.1238        0.1866      0.2416   
2     25.53      152.50  1709.0       0.1444        0.4245      0.4504   
3     26.50       98.87   567.7       0.2098        0.8663      0.6869   
4     16.67      152.20  1575.0       0.1374        0.2050      0.4000   

   concave_points3  symmetry3  fractal_dimension3  
0           0.2654     0.4601             0.11890  
1           0.1860     0.2750             0.08902  
2           0.2430     0.3613             0.08758  
3           0.2575     0.6638             0.17300  
4           0.1625     0.2364             0.07678  

[5 rows x 32 columns]
```

In [170]:
```
### Data Cleaning - Display DF features

print(df.info()) # Display features
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 32 columns):
ID                    569 non-null int64
Diagnosis             569 non-null object
radius1               569 non-null float64
texture1              569 non-null float64
perimeter1            569 non-null float64
area1                 569 non-null float64
smoothness1           569 non-null float64
compactness1          569 non-null float64
concavity1            569 non-null float64
concave_points1       569 non-null float64
symmetry1             569 non-null float64
factal_dimension1     569 non-null float64
radius2               569 non-null float64
texture2              569 non-null float64
perimeter2            569 non-null float64
area2                 569 non-null float64
smoothness2           569 non-null float64
compactness2          569 non-null float64
concavity2            569 non-null float64
concave_points2       569 non-null float64
symmetry2             569 non-null float64
fractal_dimension2    569 non-null float64
radius3               569 non-null float64
texture3              569 non-null float64
perimeter3            569 non-null float64
area3                 569 non-null float64
smoothness3           569 non-null float64
compactness3          569 non-null float64
concavity3            569 non-null float64
concave_points3       569 non-null float64
symmetry3             569 non-null float64
fractal_dimension3    569 non-null float64
dtypes: float64(30), int64(1), object(1)
memory usage: 142.3+ KB
None
```

In [171]:
```
### Data Cleaning - Display Benign Count and Malignant Count

diagnosis_counts = df['Diagnosis'].value_counts()
benign_count = diagnosis_counts['B']
malignant_count = diagnosis_counts['M']

print("Benign Count:", benign_count)
print("Malignant Count:", malignant_count)

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

```
Benign Count: 357
Malignant Count: 212
```

In [172]:
```
### Exploratory Data Analysis - display correlation matrix


df.corr()

## By observation of the correlation matrix it looks like radius, perimeter, a
nd concave points are the most significant features.
```

Out[172]:

	ID	Diagnosis	radius1	texture1	perimeter1	area1	smoothness
ID	1.000000	0.039769	0.074626	0.099770	0.073159	0.096893	-0.01296
Diagnosis	0.039769	1.000000	0.730029	0.415185	0.742636	0.708984	0.35856
radius1	0.074626	0.730029	1.000000	0.323782	0.997855	0.987357	0.17058
texture1	0.099770	0.415185	0.323782	1.000000	0.329533	0.321086	-0.02338
perimeter1	0.073159	0.742636	0.997855	0.329533	1.000000	0.986507	0.20727
area1	0.096893	0.708984	0.987357	0.321086	0.986507	1.000000	0.17702
smoothness1	-0.012968	0.358560	0.170581	-0.023389	0.207278	0.177028	1.00000
compactness1	0.000096	0.596534	0.506124	0.236702	0.556936	0.498502	0.65912
concavity1	0.050080	0.696360	0.676764	0.302418	0.716136	0.685983	0.52198
concave_points1	0.044158	0.776614	0.822529	0.293464	0.850977	0.823269	0.55369
symmetry1	-0.022114	0.330499	0.147741	0.071401	0.183027	0.151293	0.55777
factal_dimension1	-0.052511	-0.012838	-0.311631	-0.076437	-0.261477	-0.283110	0.58479
radius2	0.143048	0.567134	0.679090	0.275869	0.691765	0.732562	0.30146
texture2	-0.007526	-0.008303	-0.097317	0.386358	-0.086761	-0.066280	0.06840
perimeter2	0.137331	0.556141	0.674172	0.281673	0.693135	0.726628	0.29609
area2	0.177742	0.548236	0.735864	0.259845	0.744983	0.800086	0.24655
smoothness2	0.096781	-0.067016	-0.222600	0.006614	-0.202694	-0.166777	0.33237
compactness2	0.033961	0.292999	0.206000	0.191975	0.250744	0.212583	0.31894
concavity2	0.055239	0.253730	0.194204	0.143293	0.228082	0.207660	0.24839
concave_points2	0.078768	0.408042	0.376169	0.163851	0.407217	0.372320	0.38067
symmetry2	-0.017306	-0.006522	-0.104321	0.009127	-0.081629	-0.072497	0.20077
fractal_dimension2	0.025725	0.077972	-0.042641	0.054458	-0.005523	-0.019887	0.28360
radius3	0.082405	0.776454	0.969539	0.352573	0.969476	0.962746	0.21312
texture3	0.064720	0.456903	0.297008	0.912045	0.303038	0.287489	0.03607
perimeter3	0.079986	0.782914	0.965137	0.358040	0.970387	0.959120	0.23885
area3	0.107187	0.733825	0.941082	0.343546	0.941550	0.959213	0.20671
smoothness3	0.010338	0.421465	0.119616	0.077503	0.150549	0.123523	0.80532
compactness3	-0.002968	0.590998	0.413463	0.277830	0.455774	0.390410	0.47246
concavity3	0.023203	0.659610	0.526911	0.301025	0.563879	0.512606	0.43492
concave_points3	0.035174	0.793566	0.744214	0.295316	0.771241	0.722017	0.50305
symmetry3	-0.044224	0.416294	0.163953	0.105008	0.189115	0.143570	0.39430
fractal_dimension3	-0.029866	0.323872	0.007066	0.119205	0.051019	0.003738	0.49931

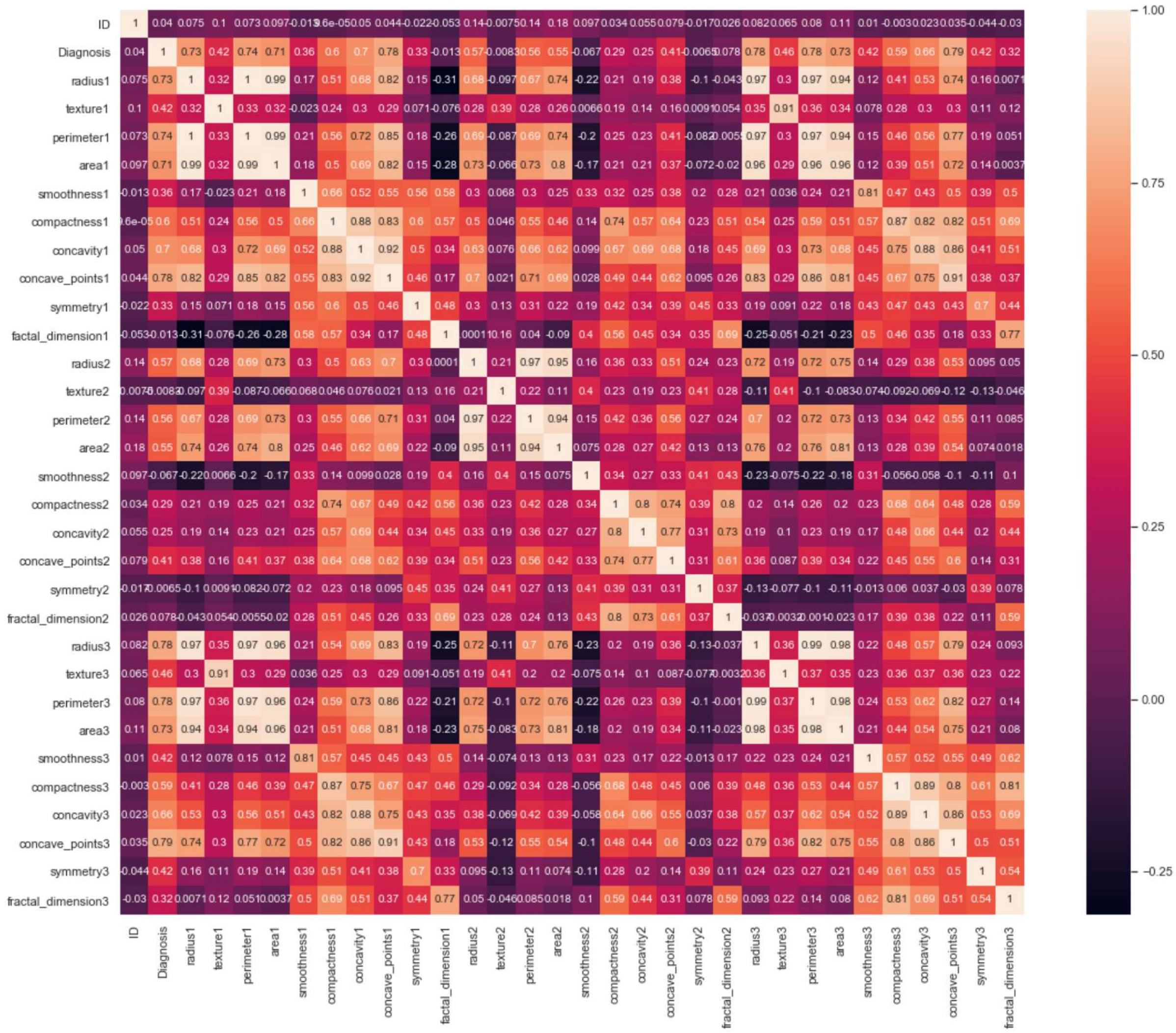
32 rows × 32 columns

In [173]:

```python
### Exploratory Data Analysis - display heatmap matrix
# Looking at the heatmap it also appears the following features are colinear
# due to their high correlation values: radius1 perimeter1, area1, radius3, pe
rimeter3, and area3.

plt.subplots(figsize=(20,15))
sns.heatmap(df.corr(), annot=True, square=True, xticklabels=True, cbar=True)
```

Out[173]: <matplotlib.axes._subplots.AxesSubplot at 0x1b0d4b56550>



In [174]:
```
### Models - Strongest Indicator
# Using only concave_points3 the strongest indicator as the only indicator we
can get an R-squared value of 0.72

X = df.drop('Diagnosis', axis=1)
y = df['Diagnosis']

x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
best_r_squared = 0
max_deg = 10
formula = 'Diagnosis ~ concave_points3'
model = smf.ols(formula, data=df).fit()

print("Degree:  1",  model.rsquared)

for deg in range(2,max_deg +1):
    formula += f' + np.power(concave_points3,{deg})'
    model = smf.ols(formula, data=df).fit()
    print("Degree: ", deg, model.rsquared)
    if(model.rsquared_adj > best_r_squared):
        best_formula = formula
        best_model = model
        best_r_squared = best_model.rsquared_adj
        best_degree = deg

formula_strongest_ind = 'Diagnosis ~ concave_points3 + np.power(concave_points
3,2) + np.power(concave_points3, 3)'
```

```
Degree:  1 0.6297470235614584
Degree:  2 0.6309352699330446
Degree:  3 0.7033610378146713
Degree:  4 0.7033900659916137
Degree:  5 0.7202997951788024
Degree:  6 0.7204212156055667
Degree:  7 0.7242914692809307
Degree:  8 0.7242967644841682
Degree:  9 0.725312356224284
Degree:  10 0.7253858093393647
```

In [175]:
```
### Models - Multilinear
# Using backwards elimination of features and additional volume features, we c
an get an R-squared value of 0.80
# Added volume features that may be impactful.

df['AR1'] = df['radius1']*df['area1']
df['AR2'] = df['radius2']*df['area2']
df['AR3'] = df['radius3']*df['area3']
df['R1cube'] = df['radius1']*df['radius1']*df['radius1']
df['R2cube'] = df['radius2']*df['radius2']*df['radius2']
df['R3cube'] = df['radius3']*df['radius3']*df['radius3']
df['strong_ind'] = np.power(df['concave_points3'], 3)

features = list(df.columns)
features.remove('Diagnosis')
features.remove('ID')

formula = f"Diagnosis ~ {' + '.join(features)}"
psiglevel = 0.01
while True:
    model = smf.ols(formula, data=df).fit()
    max_pvalue = model.pvalues[1:].max()
    if max_pvalue > psiglevel and model.pvalues[1:].idxmax()[:] != 'Intercep
t':
        feature_to_remove = model.pvalues[1:].idxmax()[:]
        features.remove(feature_to_remove)
        formula = f"Diagnosis ~ {' + '.join(features)}"
    else:
        break

Backwards_Formula = formula
print("Backwards Formula:", Backwards_Formula)
print("Rsquared:", model.summary())
print("Rsquared:", model.rsquared)
```

```
Backwards Formula: Diagnosis ~ perimeter1 + area1 + concavity1 + smoothness2
+ concavity2 + concave_points2 + radius3 + texture3 + compactness3 + symmetry
3 + AR1 + R3cube
Rsquared:                            OLS Regression Results
==============================================================================
=
Dep. Variable:              Diagnosis   R-squared:                       0.80
0
Model:                            OLS   Adj. R-squared:                  0.79
6
Method:                 Least Squares   F-statistic:                     185.
4
Date:                Tue, 12 Dec 2023   Prob (F-statistic):          2.10e-18
5
Time:                        23:19:56   Log-Likelihood:                 64.12
1
No. Observations:                 569   AIC:                            -102.
2
Df Residuals:                     556   BIC:                            -45.7
7
Df Model:                          12
Covariance Type:            nonrobust
==============================================================================
======
                      coef    std err          t      P>|t|      [0.025
0.975]
------------------------------------------------------------------------------
------
Intercept           0.2941      0.281      1.046      0.296      -0.258
0.846
perimeter1         -0.0741      0.007    -10.243      0.000      -0.088
-0.060
area1               0.0074      0.001      7.117      0.000       0.005
0.009
concavity1          3.6420      0.394      9.247      0.000       2.868
4.416
smoothness2        10.9796      3.957      2.775      0.006       3.208
18.752
concavity2         -5.6529      0.680     -8.311      0.000      -6.989
-4.317
concave_points2    13.7531      2.874      4.785      0.000       8.108
19.398
radius3             0.1708      0.022      7.861      0.000       0.128
0.213
texture3            0.0092      0.002      5.420      0.000       0.006
0.013
compactness3        0.5010      0.125      3.999      0.000       0.255
0.747
symmetry3           0.7909      0.201      3.937      0.000       0.396
1.185
AR1                -0.0001   2.29e-05     -4.659      0.000      -0.000   -6.
16e-05
R3cube             -0.0001   1.15e-05     -8.862      0.000      -0.000   -7.
94e-05
==============================================================================
=
Omnibus:                       55.209   Durbin-Watson:                   1.73
```

```
9
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               75.01
5
Skew:                           0.730   Prob(JB):                     5.14e-1
7
Kurtosis:                       4.017   Cond. No.                     7.35e+0
6
=============================================================================
=

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correc
tly specified.
[2] The condition number is large, 7.35e+06. This might indicate that there a
re
strong multicollinearity or other numerical problems.
Rsquared: 0.800076491433471
```

In [176]:
```
### Models - Strongest Indicators
# Using only concave_points3 the strongest indicator as the only indicator we
can get an R-squared value of 0.74

features = list(df.columns)
features.remove('Diagnosis')
features.remove('ID')

best = ['',0]
best_features = []
x_train, x_test = train_test_split(df, train_size=0.2, random_state=10)

for p in features:
    model  = smf.ols(formula='Diagnosis~'+p, data=x_train).fit()
    if model.rsquared>best[1]:
        best = [p, model.rsquared]

best_features.append(best[0])
features.remove(best[0])

for i in range(4):

    for p in features:
        formula = f'Diagnosis ~ {" + ".join(best_features)} + {p}'
        model  = smf.ols(formula, data=x_train).fit()
        if model.rsquared>best[1]:
            best = [p, model.rsquared]
    best_features.append(best[0])
    features.remove(best[0])



formula_best_features = f'Diagnosis ~ {" + ".join(best_features)} + {p}'
print("Best Featues formula: ", formula_best_features )
print("Rsquared:", model.summary())
print("Rsquared:", model.rsquared)
```

```
Best Featues formula:  Diagnosis ~ concave_points3 + texture3 + radius3 + R3c
ube + smoothness3 + strong_ind
Rsquared:                              OLS Regression Results
==============================================================================
=
Dep. Variable:            Diagnosis   R-squared:                       0.74
9
Model:                          OLS   Adj. R-squared:                  0.73
7
Method:               Least Squares   F-statistic:                     63.7
2
Date:              Tue, 12 Dec 2023   Prob (F-statistic):          1.66e-3
0
Time:                      23:19:57   Log-Likelihood:                -1.170
8
No. Observations:               113   AIC:                             14.3
4
Df Residuals:                   107   BIC:                             30.7
1
Df Model:                         5
Covariance Type:          nonrobust
==============================================================================
======
                      coef    std err          t      P>|t|      [0.025
0.975]
------------------------------------------------------------------------------
------
Intercept          -1.5602      0.217     -7.206      0.000      -1.989
-1.131
concave_points3     3.0656      1.086      2.822      0.006       0.912
5.219
texture3            0.0171      0.004      4.162      0.000       0.009
0.025
radius3             0.0802      0.018      4.419      0.000       0.044
0.116
R3cube          -3.939e-05   1.18e-05     -3.333      0.001   -6.28e-05      -
1.6e-05
strong_ind         10.9234     14.662      0.745      0.458     -18.143
39.990
==============================================================================
=
Omnibus:                      3.015   Durbin-Watson:                   2.24
2
Prob(Omnibus):                0.221   Jarque-Bera (JB):                2.99
7
Skew:                         0.356   Prob(JB):                        0.22
4
Kurtosis:                     2.638   Cond. No.                     5.67e+0
6
==============================================================================
=

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correc
tly specified.
[2] The condition number is large, 5.67e+06. This might indicate that there a
re
```

```
strong multicollinearity or other numerical problems.
Rsquared: 0.7485915336935886
```

In [178]:
```
#  Results
x_train, x_test = train_test_split(df, train_size=0.2, random_state=10)


train_model_strongest_ind = smf.ols(formula=formula_strongest_ind, data=x_trai
n).fit()
model_strongest_ind = smf.ols(formula=formula_strongest_ind, data=x_test).fit
()
y_pred_strongest_ind = train_model_strongest_ind.predict(x_test)
fpr_strongest_ind, tpr_strongest_ind, _ = roc_curve(x_test['Diagnosis'], y_pre
d_strongest_ind)
roc_auc_strongest_ind = auc(fpr_strongest_ind, tpr_strongest_ind)
y_pred_binary_strongest_ind = (y_pred_strongest_ind > 0.5).astype(int)
f1_strongest_ind = f1_score(x_test['Diagnosis'], y_pred_binary_strongest_ind)

print("Strongest Ind - formula: ", formula_strongest_ind)
print("Strongest Ind - ADJ Rsquared:", model_strongest_ind.rsquared_adj)
print("Strongest Ind - ROC-AUC Strongest Ind:", roc_auc_strongest_ind)
print("Strongest Ind - F1 Score Strongest Ind:", f1_strongest_ind)





train_model_backwards = smf.ols(formula=Backwards_Formula, data=x_train).fit()
model_backwards = smf.ols(formula=Backwards_Formula, data=x_test).fit()
y_pred_backwards = train_model_backwards.predict(x_test)
fpr_backwards, tpr_backwards, _ = roc_curve(x_test['Diagnosis'], y_pred_backwa
rds)
roc_auc_backwards = auc(fpr_backwards, tpr_backwards)
y_pred_binary_backwards = (y_pred_backwards > 0.5).astype(int)
f1_backwards = f1_score(x_test['Diagnosis'], y_pred_binary_backwards)

print ('\n')
print("Backwards Refinement - formula: ", Backwards_Formula)
print("Backwards Refinement - ADJ Rsquared:", model_backwards.rsquared_adj)
print("Backwards Refinement - ROC-AUC Backwards:", roc_auc_backwards)
print("Backwards Refinement - F1 Score Backwards:", f1_backwards)





train_model_best_features = smf.ols(formula=formula_best_features, data=x_trai
n).fit()
model_best_features = smf.ols(formula=formula_best_features, data=x_test).fit
()
y_pred_best_features = train_model_best_features.predict(x_test)
fpr_best_features, tpr_best_features, _ = roc_curve(x_test['Diagnosis'], y_pre
d_best_features)
roc_auc_best_features = auc(fpr_best_features, tpr_best_features)
y_pred_binary_best_features = (y_pred_best_features > 0.5).astype(int)
f1_best_features = f1_score(x_test['Diagnosis'], y_pred_binary_best_features)


print ('\n')
print("Forward Refinement - formula: ", formula_best_features)
```

```
print("Forward Refinement - ADJ Rsquared:", model_best_features.rsquared_adj)
print("Forward Refinement - ROC-AUC Best Features:", roc_auc_best_features)
print("Forward Refinement - F1 Score Best Features:", f1_best_features)
```

```
Strongest Ind - formula:  Diagnosis ~ concave_points3 + np.power(concave_poin
ts3,2) + np.power(concave_points3, 3)
Strongest Ind - ADJ Rsquared: 0.6978324367912714
Strongest Ind - ROC-AUC Strongest Ind: 0.960978835978836
Strongest Ind - F1 Score Strongest Ind: 0.8711656441717791


Backwards Refinement - formula:  Diagnosis ~ perimeter1 + area1 + concavity1
+ smoothness2 + concavity2 + concave_points2 + radius3 + texture3 + compactne
ss3 + symmetry3 + AR1 + R3cube
Backwards Refinement - ADJ Rsquared: 0.7915645212265836
Backwards Refinement - ROC-AUC Backwards: 0.9954117063492064
Backwards Refinement - F1 Score Backwards: 0.9473684210526314


Forward Refinement - formula:  Diagnosis ~ concave_points3 + texture3 + radiu
s3 + R3cube + smoothness3 + strong_ind
Forward Refinement - ADJ Rsquared: 0.7402375060491297
Forward Refinement - ROC-AUC Best Features: 0.9908027447089948
Forward Refinement - F1 Score Best Features: 0.9415384615384617
```

In [ ]:

```
## Discussion and Conclusion

# Best r_squared value value is backwards refinement. It looks like even just
using the polynomial concatenation of the
# strongest indicator results in a 0.96 ROC AUC

# Best formula was using the backwards refinement which resulted in an adjuste
d R2 of 0.79 and an ROC-AUC curve of 0.995

# Some key take-aways is that with even just the strongest indicator you can g
et some accurate classification of a cancerous tumor.
# In the future, this can generate an even more accurate model using a neural
network.
```

In [ ]:

In [ ]: