RATIONAL STATEMENT

Mr. John Hughes needs a model to determine tumor, and has asked we develop three models (Support Vector Machine (SVM), Logistic Regression, and Decision Tree). This will help him accurately conduct patient diagnosis that will determine if they are Benign or Malignant tumor.

METHODOLOGY

We used three algorithms (SVM, Logistic Regression, and Decision Tree) to carry out the analysis. Based on the outputs, we will see which model is best for conducting, predicting and determining patients who have Benignant or Malignant tumor.

BASIC STATISTICS KEY INSIGHTS

	count	mean	std	min	25%	50%	75%	max
radius_mean	569.0	14.127292	3.524049	6.981000	11.700000	13.370000	15.780000	28.11000
texture_mean	569.0	19.289649	4.301036	9.710000	16.170000	18.840000	21.800000	39.28000
perimeter_mean	569.0	91.969033	24.298981	43.790000	75.170000	86.240000	104.100000	188.50000
area_mean	569.0	654.889104	351.914129	143.500000	420.300000	551.100000	782.700000	2501.00000

The size (**count)** of our data is pretty good at 569.0, anything below sample size of 30 would have been a call for concern. Giving us a good sample size to build a model.

Secondly, looking at the spread (**standard deviations**) are well and visibly all over the place for each independent variable.

With regards to our quartiles (**25%, 50%, 75%**), there is not much of a similarity, rather the difference is clear, an increase is noticed at each percentile. Hence value count is not close

The range (**minimum and maximum or min & max**) for some of the variables is huge, spread, showing a significant difference.

BASIC STATISTICS KEY INSIGHTS (continuation)

	count	mean	std	min	25%	50%	75%	max
radius_mean	569.0	14.127292	3.524049	6.981000	11.700000	13.370000	15.780000	28.11000
texture_mean	569.0	19.289649	4.301036	9.710000	16.170000	18.840000	21.800000	39.28000
perimeter_mean	569.0	91.969033	24.298981	43.790000	75.170000	86.240000	104.100000	188.50000
area_mean	569.0	654.889104	351.914129	143.500000	420.300000	551.100000	782.700000	2501.00000

Huge outliers as the spread is large

Looking at our central point (**mean**)

On average each independent variable when compared to the other gives up some points hence not really predicting well, looking at it from (radius_mean to area_mean). Each mean is larger than the previous.

The averages are rightly skewed from the max

KEY INSIGHTS FROM ALGORITHMS (SVM)

```
#SCRIPT FOR SVM
from sklearn.svm import SVC
from sklearn.metrics import classification_report, confusion_matrix


for name,method in [('SVM', SVC(kernel='linear',random_state=100))]:
    method.fit(x_train2,y_train)
    predict = method.predict(x_test2)
    print(confusion_matrix(y_test,predict))
    print(classification_report(y_test,predictions))
```

```
[[70  2]
 [ 3 39]]
              precision    recall  f1-score   support

           B       0.96      0.97      0.97        72
           M       0.95      0.93      0.94        42

    accuracy                           0.96       114
   macro avg       0.96      0.95      0.95       114
weighted avg       0.96      0.96      0.96       114
```

Top left output is confusion matrix , below it we have the classification report done together. It will be wrong to omit the former two as they help ensure correct calculations.

Support says it is a Large dataset of one hundred and fourteen(114), but more of one class (B) than another (M).

Very similar metrics

from a **Precision** point of view at 0.96, how precise it is at predicting patients who have benignant(B) or malignant(M) tumor. Precision does a very good job at 0.96

Recall at 0.96, how many times can it be predicted the patients are benignant(B) or malignant(M). So high at 0.96, a pretty good reoccurrence.

F1-Score at 0.96, is the weighted average of precision and f1-score which is still good

Since this is an imbalance dataset, I look at **recall** and **f1-score** which shows this is a pretty good model to use.

Top half, how good is it at predicting patients who have and do not have tumor. I will say very good at 0.96 for B and 0.95 for M. Should I implement this model for predicting and not predicting tumor? Yes, it should be used for getting both results.

KEY INSIGHTS FROM ALGORITHMS (LOGISTICAL REGRESSION)

```
#SCRIPT FOR LOGISTICAL REGRESSION
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix

for name,method in [('Logistic Regression', LogisticRegression(solver='liblinear',random_state=100))]:
    method.fit(x_train2,y_train)
    predict = method.predict(x_test2)
    print(confusion_matrix(y_test,predict))
    print(classification_report(y_test,predict))
```

```
[[70  2]
 [ 1 41]]
              precision    recall  f1-score   support

           B       0.99      0.97      0.98        72
           M       0.95      0.98      0.96        42

    accuracy                           0.97       114
   macro avg       0.97      0.97      0.97       114
weighted avg       0.97      0.97      0.97       114
```

Top left output is confusion matrix , below it we have the classification report done together. It will be wrong to omit the former two as they help ensure correct calculations.

Support says it is a Large dataset of one hundred and fourteen(114), but more of one class (B) than another (M).

Same metrics

from a **Precision** point of view at 0.97, how precise it is at predicting patients who have benignant(B) or malignant(M) tumor. Very precise at 0.97 without a doubt.

Recall at 0.97, how many times can it be predicted the patients are benignant(B) or malignant(M). So high at 0.97, a very good reoccurrence.

F1-Score at 0.97, is the weighted average of precision and f1-score which is good.

Since this is an imbalance dataset, I will look at **recall** and **f1-score** which shows this is highly a good model to use.

Going by both classes (M & B), how good is it at predicting patients who have and do not have tumor. This model is very good at 0.99 for B and 0.95 for M. Should I implement this model for predicting and not predicting tumor? Yes, it should be used for getting both results, especially for B which is almost 100% certain.

KEY INSIGHTS FROM ALGORITHMS (DECISION TREE)

```
#SCRIPT FOR DECISION TREE
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report, confusion_matrix

for name,method in [('DT', DecisionTreeClassifier(random_state=100))]:
    method.fit(x_train2,y_train)
    predict = method.predict(x_test2)
    print('\nEstimator: {}'.format(name))
    print(confusion_matrix(y_test,predict))
    print(classification_report(y_test,predict))
```

```
Estimator: DT
[[67  5]
 [ 4 38]]
              precision    recall  f1-score   support

           B       0.94      0.93      0.94        72
           M       0.88      0.90      0.89        42

    accuracy                           0.92       114
   macro avg       0.91      0.92      0.92       114
weighted avg       0.92      0.92      0.92       114
```

Top left output is confusion matrix , below it we have the classification report done together. It will be wrong to omit the former two as they help ensure correct calculations.

Support says it is a Large dataset of one hundred and fourteen(114), but more of one class (B) than another (M).

Similar metrics

Looking at **Precision** with 0.92, how precise it is at predicting patients who have benignant(B) or malignant(M) tumor. Precision does a very precise prediction at 0.92.

Recall at 0.92, that is if the patients want the test redone, how many times can it be predicted the patients are benignant(B) or malignant(M). So high at 0.92, a pretty good reoccurrence.

F1-Score at 0.92, is the weighted average of precision and f1-score which is still a good prediction.

Since this is an imbalance dataset, I look at **recall** and **f1-score** which shows this is a pretty good model to use.

With both tumors, how good is it at predicting both benignant and Malignant. I will say good at 0.94 for B and 0.88 for M. Should I implement this model for predicting and not predicting tumor? Yes, it should be used for getting both results, but better for predicting B.

MODEL RECOMMENDATION

The **logistic algorithm** to me did a pretty good job with the dataset. I will recommend this model for it's;

- Accuracy
- Weighted average
- Predictability
- Recall and f1-score

When compared with support vector machine (SVM), the logistical regression algorithm is more accurate and precise for both B and M.

```
#SCRIPT FOR LOGISTICAL REGRESSION
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix

for name,method in [('Logistic Regression', LogisticRegression(solver='liblinear',random_state=100))]:
    method.fit(x_train2,y_train)
    predict = method.predict(x_test2)
    print(confusion_matrix(y_test,predict))
    print(classification_report(y_test,predict))
```

```
[[70  2]
 [ 1 41]]
              precision    recall  f1-score   support

           B       0.99      0.97      0.98        72
           M       0.95      0.98      0.96        42

    accuracy                           0.97       114
   macro avg       0.97      0.97      0.97       114
weighted avg       0.97      0.97      0.97       114
```

POSSIBLE IMPROVEMENTS FOR LOGISTIC MODEL

Take a proper look at the independent variables to assure they are the right variables. Is it possible to use other variables

Find a way to balance the dataset, since it’s an imbalance dataset