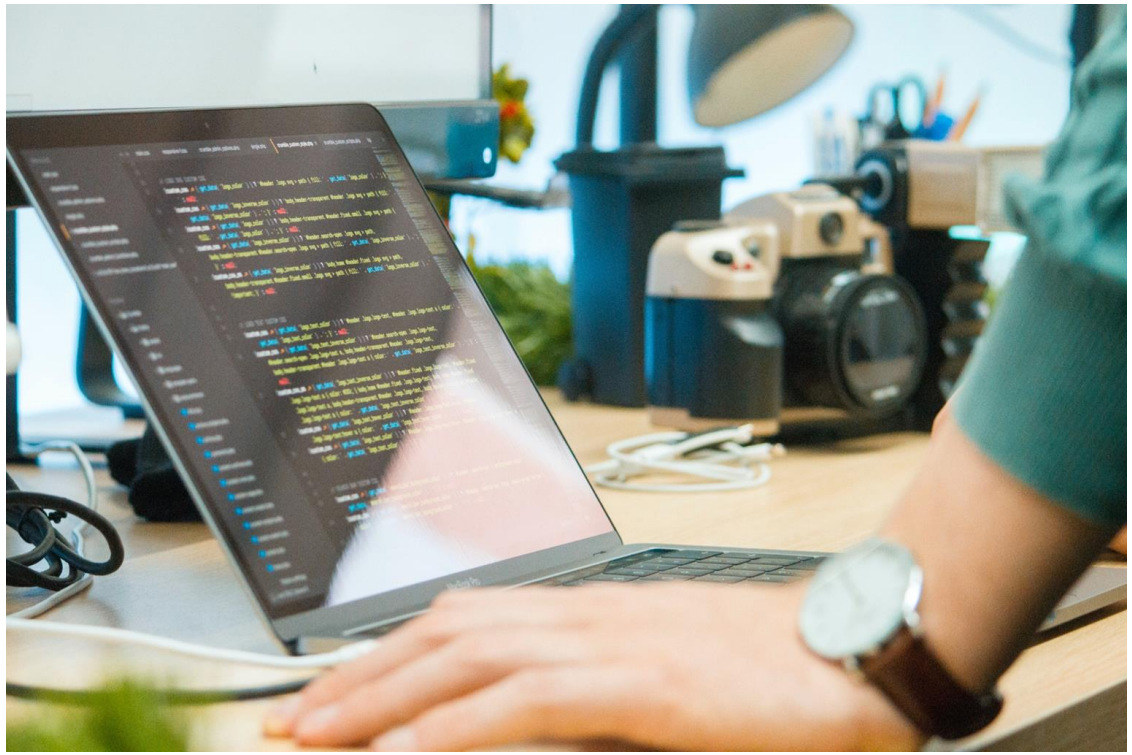


Final Project
Breast Cancer Wisconsin Classifer

Team Alpha

Indonesia AI
AI for Everyone, AI for Indonesia

1. Amira Amandanisa - Data Loading, Data Visualization, Data preprocessing
2. Septian Maulana Yusuf - SVM, Decision Tree
3. Ricky Alexander Bianco - Splitting Data, Random Forest

Background

Background

Indonesia AI
AI for Everyone, AI for Indonesia

Dataset ini mengenai kanker payudara yang didapat melalui perhitungan terhadap gambar digital atas uji Aspirasi Jarum Halus(FNA) dari massa pyudara. Data ini menggambarkan karakteristik dari inti sel.

- **Tujuan**

  - ❖ Mengidentifikasi jumlah kelas kanker jinak (benign) atau ganas(maligant)

Request

- Membuat AI yang dapat mengklasifikasi kanker dengan tipe jinak(benign) dan ganas(maligant)
- Melakukan teknik eksplorasi dan visualisasi data
- Membuat AI dengan kemampuan klasifikasi menggunakan minimal 2 model yang telah dipelajari dengan F1 score diatas 0.75

Data Understanding

Indonesia AI
AI for Everyone, AI for Indonesia

Load CSV to DataFrame

Indonesia AI
AI for Everyone, AI for Indonesia

```
df = pd.read_csv('breastcancer.csv')
```

```
df.head()
```

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	...
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	...
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	...
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	...
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	...
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	...

5 rows × 33 columns

Data Shape, Type, Non Null

Indonesia AI
AI for Everyone, AI for Indonesia

```
5]  #Check size
    df.shape
```

```
(569, 33)
```

```
#Check duplicate sum
df.duplicated().sum()
```

```
0
```

```
#Check Null Value
df.isnull().sum()
```

```
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
Unnamed: 32              569
dtype: int64
```

```
#Check Missing Value
df.isna().sum()
```

```
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
Unnamed: 32              569
dtype: int64
```

Data Info

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 33 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   id                       569 non-null    int64  
 1   diagnosis                569 non-null    object 
 2   radius_mean              569 non-null    float64
 3   texture_mean             569 non-null    float64
 4   perimeter_mean           569 non-null    float64
 5   area_mean                569 non-null    float64
 6   smoothness_mean          569 non-null    float64
 7   compactness_mean         569 non-null    float64
 8   concavity_mean           569 non-null    float64
 9   concave points_mean      569 non-null    float64
 10  symmetry_mean            569 non-null    float64
 11  fractal_dimension_mean   569 non-null    float64
 12  radius_se                569 non-null    float64
 13  texture_se               569 non-null    float64
 14  perimeter_se             569 non-null    float64
 15  area_se                  569 non-null    float64
 16  smoothness_se            569 non-null    float64
 17  compactness_se           569 non-null    float64
 18  concavity_se             569 non-null    float64
 19  concave points_se        569 non-null    float64
 20  symmetry_se              569 non-null    float64
 21  fractal_dimension_se     569 non-null    float64
 22  radius_worst             569 non-null    float64
 23  texture_worst            569 non-null    float64
 24  perimeter_worst          569 non-null    float64
 25  area_worst               569 non-null    float64
 26  smoothness_worst         569 non-null    float64
 27  compactness_worst        569 non-null    float64
 28  concavity_worst          569 non-null    float64
 29  concave points_worst     569 non-null    float64
 30  symmetry_worst           569 non-null    float64
 31  fractal_dimension_worst  569 non-null    float64
 32  Unnamed: 32              0 non-null      float64
dtypes: float64(31), int64(1), object(1)
memory usage: 146.8+ KB
```

The Breast Cancer (Wisconsin) Diagnosis dataset contains the diagnosis and a set of 30 features describing the characteristics of the cell nuclei present in the digitized image of a of a fine needle aspirate (FNA) of a breast mass.

Ten real-valued features are computed for each cell nucleus:

- radius (mean of distances from center to points on the perimeter);
- texture (standard deviation of gray-scale values);
- perimeter;
- area;
- smoothness (local variation in radius lengths);
- compactness (perimeter^2 / area - 1.0);
- concavity (severity of concave portions of the contour);
- concave points (number of concave portions of the contour);
- symmetry;
- fractal dimension (“coastline approximation” - 1).

The mean, standard error (SE) and “worst” or largest (mean of the three largest values) of these features were computed for each image, resulting in 30 features.

Exploratory Data Analysis

EDA

Indonesia AI
AI for Everyone, AI for Indonesia

```
#Drop the column with all missing values (na, NAN, N
#NOTE: This drops the column Unnamed
df = df.dropna(axis=1)

#Get the new count of the number of rows and cols
df.shape
```

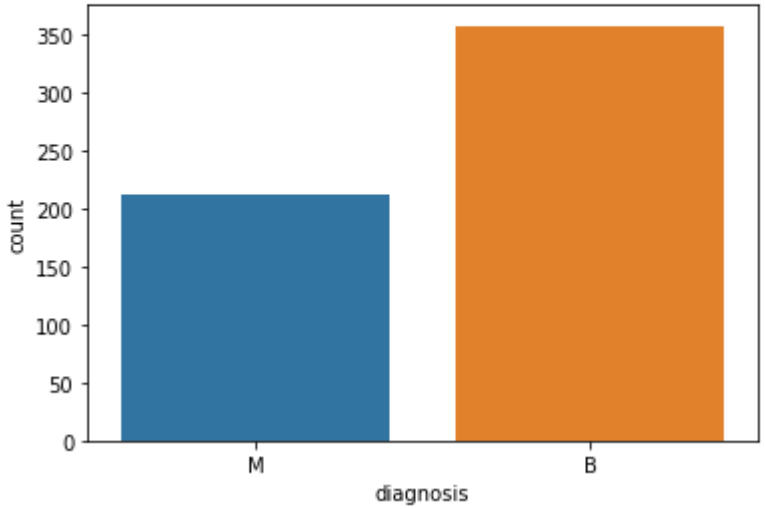
```
(569, 32)
```

```
#Get a count of the number of 'M' & 'B' cells
df['diagnosis'].value_counts()
```

```
B    357
M    212
Name: diagnosis, dtype: int64
```

```
#Visualize this count
sns.countplot(df['diagnosis'],label="Count")
```

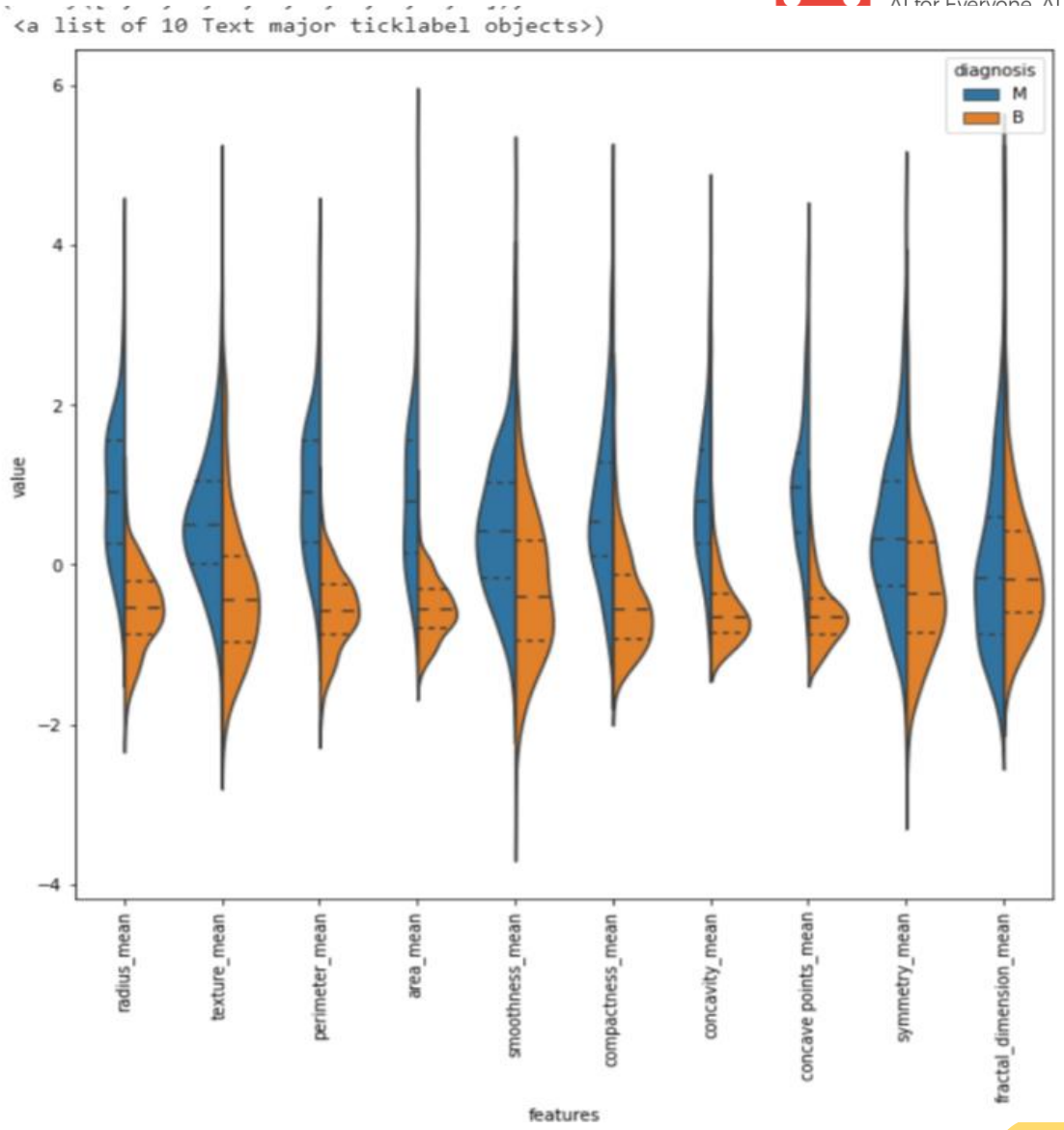
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f4dc1df5310>
```



Data Visualization

```
# first ten features
data_dia = y
data = X
data_std = (data - data.mean()) / (data.std())              # standardization
data = pd.concat([y,data_std.iloc[:,0:10]],axis=1)
data = pd.melt(data,id_vars="diagnosis",
                    var_name="features",
                    value_name='value')
plt.figure(figsize=(10,10))
sns.violinplot(x="features", y="value", hue="diagnosis", data=data,split=True, inner="quart")
plt.xticks(rotation=90)
```

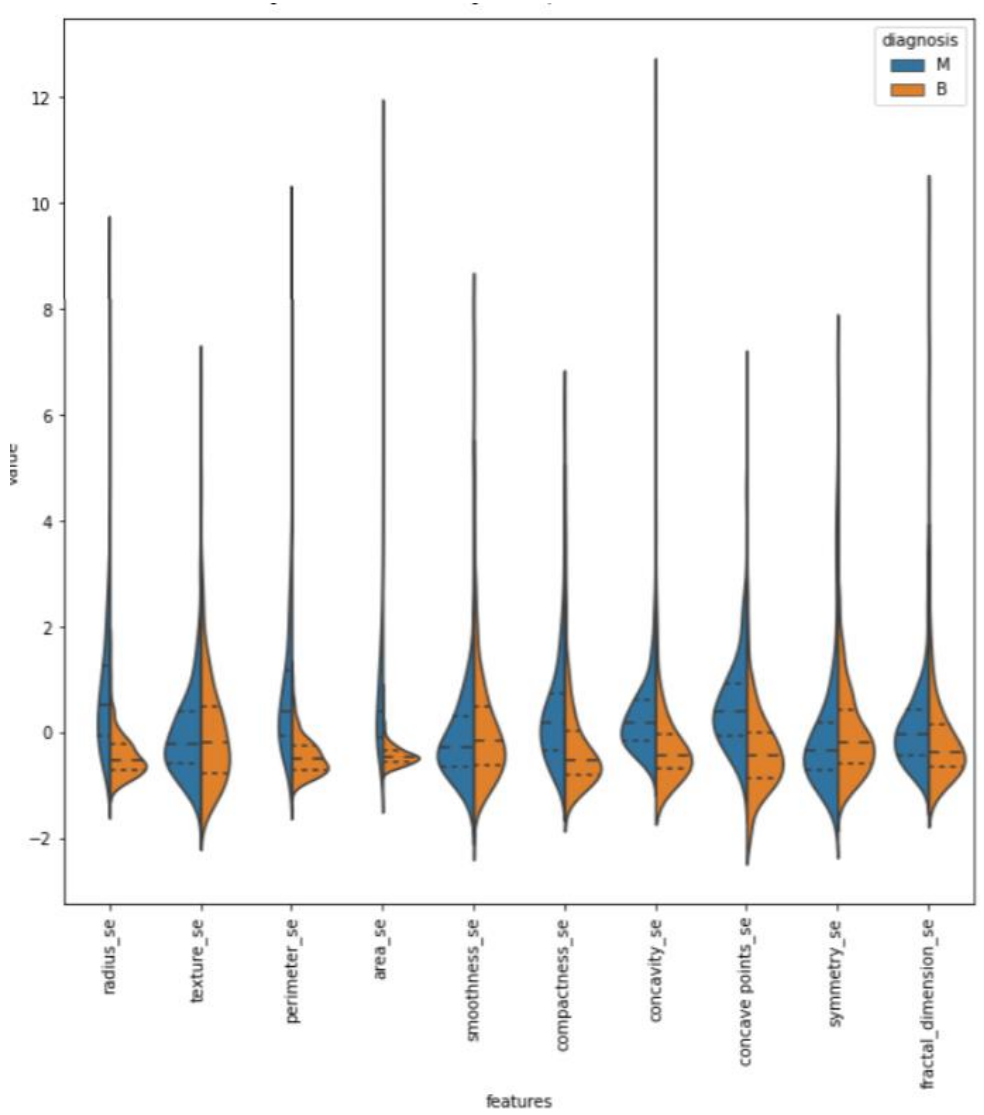
- texture_mean untuk malignant dan benign terlihat terpisah. Fitur ini mungkin bagus untuk klasifikasi
- fractal_dimension_mean, median antara malignant dan benign berdekatan satu sama lain.



Data Visualization

Indonesia AI
AI for Everyone, AI for Indonesia

```
20] # Second ten features
    data = pd.concat([y,data_std.iloc[:,10:20]],axis=1)
    data = pd.melt(data,id_vars="diagnosis",
                        var_name="features",
                        value_name='value')
    plt.figure(figsize=(10,10))
    sns.violinplot(x="features", y="value", hue="diagnosis", data=data,split=True, inner="quart")
    plt.xticks(rotation=90)
```

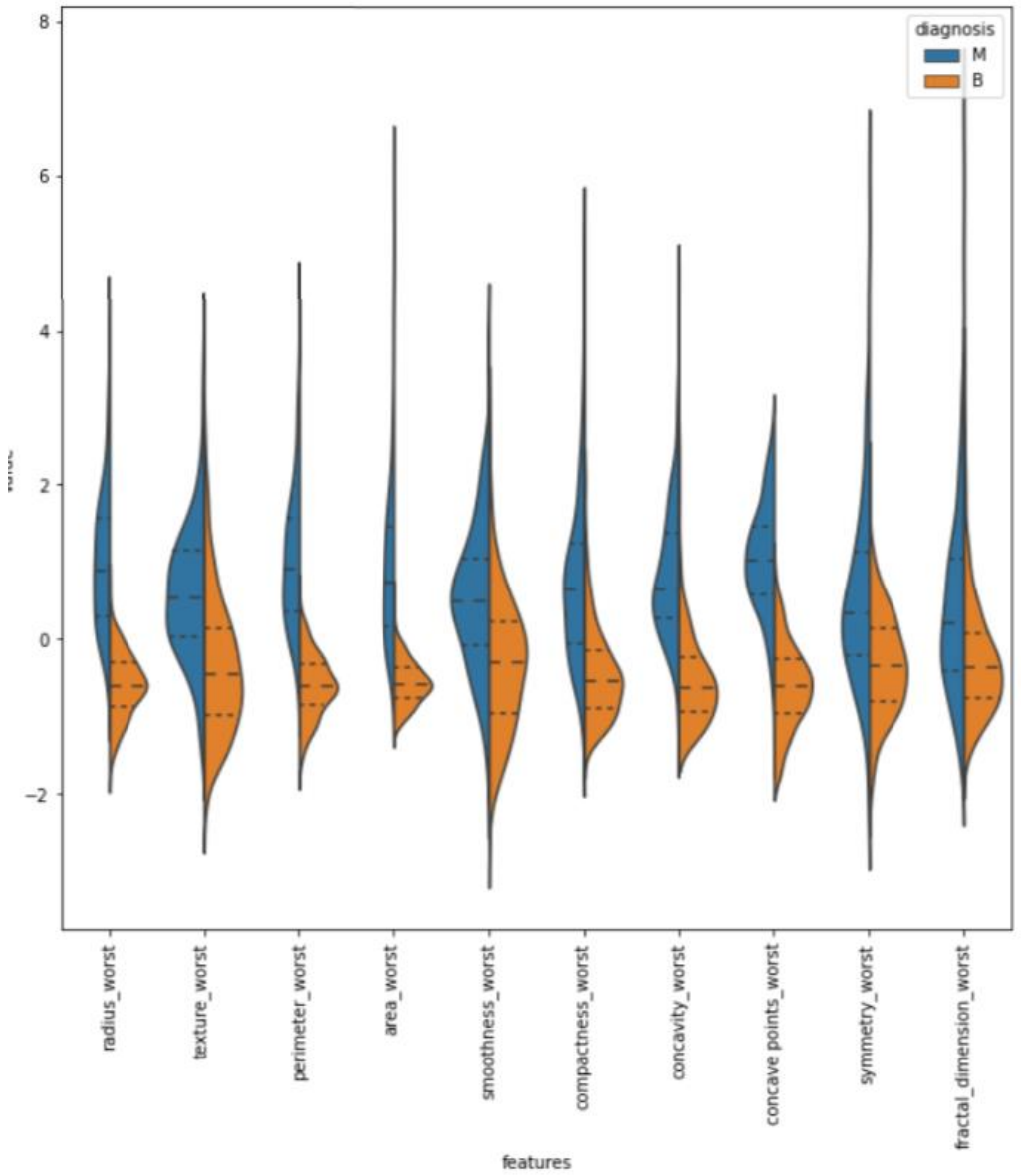


- Semua fitur pada violin plot untuk Malignant dan Benign tidak berbeda jauh
- Bentuk violin plot pada area_se terlihat sempit, sehingga distibusi sulit terlihat

Data Visualization

```
] # Last ten features
  data = pd.concat([y,data_std.iloc[:,20:31]],axis=1)
  data = pd.melt(data,id_vars="diagnosis",
                      var_name="features",
                      value_name='value')
  plt.figure(figsize=(10,10))
  sns.violinplot(x="features", y="value", hue="diagnosis", data=data,split=True, inner="quart")
  plt.xticks(rotation=90)
```

- area_worst terlihat terpisah dengan baik, fitur ini mungkin mudah untuk digunakan klasifikasi
- Pada fractal_dimension_worst varians terlihat tinggi
- Concavity_worst dan concave_points_worts memiliki distribusi data yang mirip.

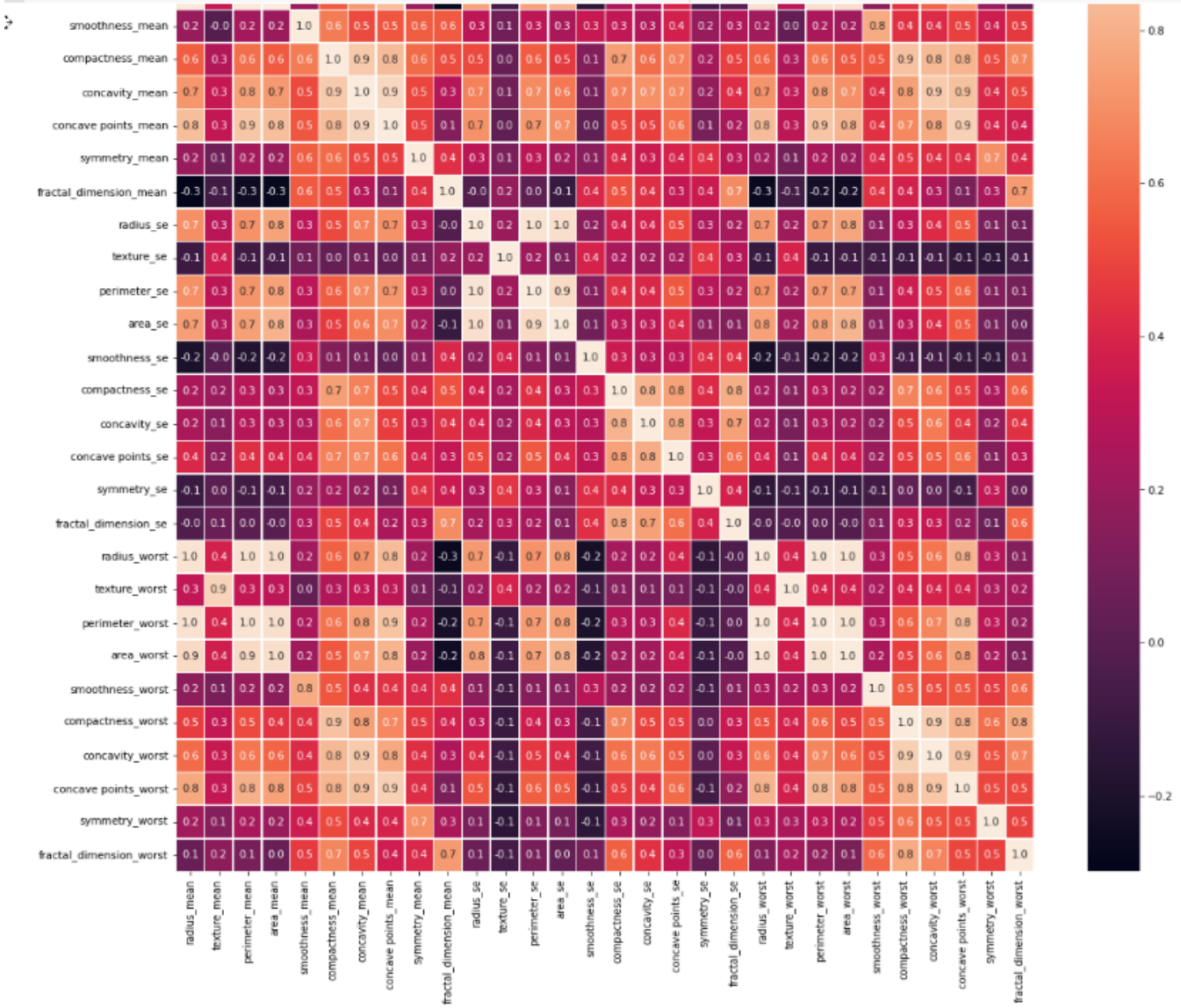


Data Visualization: Heatmap

Indonesia AI
AI for Everyone, AI for Indonesia

- Radius, parameter dan area berkorelasi kuat
- Compactness_mean, concativity_mean dan concavepoint_mean berkorelasi kuat. Harus dipilih salah satu.
- Banyak fitur yang memiliki korelasi kuat

```
#correlation map
f,ax = plt.subplots(figsize=(18, 18))
sns.heatmap(x_train_new.corr(), annot=True, linewidths=.5, fmt= '.1f',ax=ax)
```

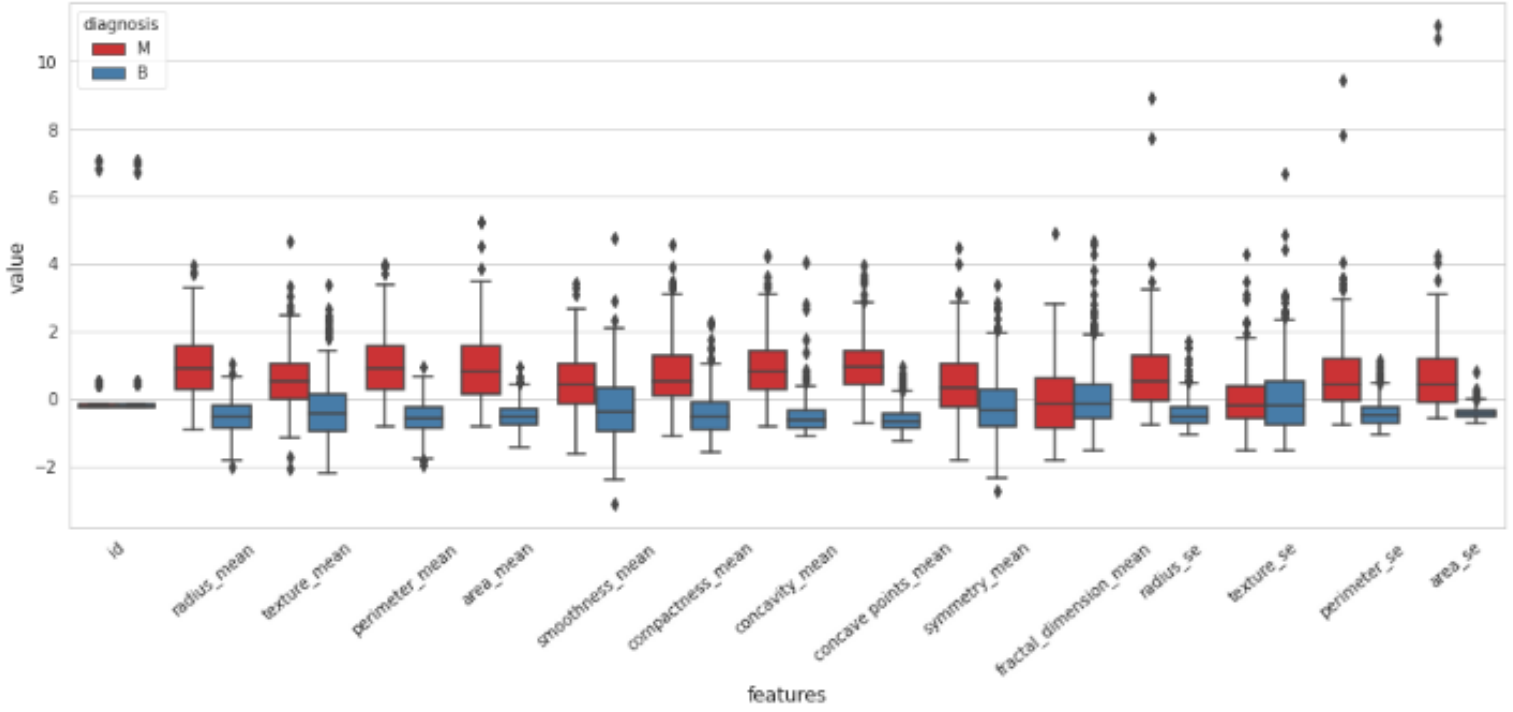


Outlier Checking

Indonesia AI
AI for Everyone, AI for Indonesia

```
plt.style.use('ggplot')
sns.set_style('whitegrid')
plt.figure(figsize=(16,6))
sns.boxplot(x="features", y="value", hue="diagnosis", data=data,palette='Set1')
plt.xticks(rotation=40)
```

```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14]),
 <a list of 15 Text major ticklabel objects>)
```

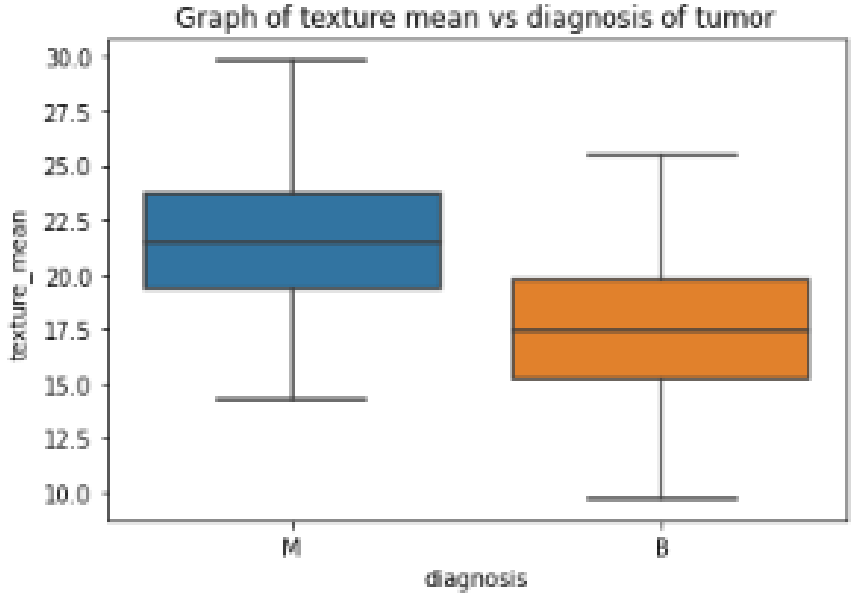


Outlier Checking

Indonesia AI
AI for Everyone, AI for Indonesia

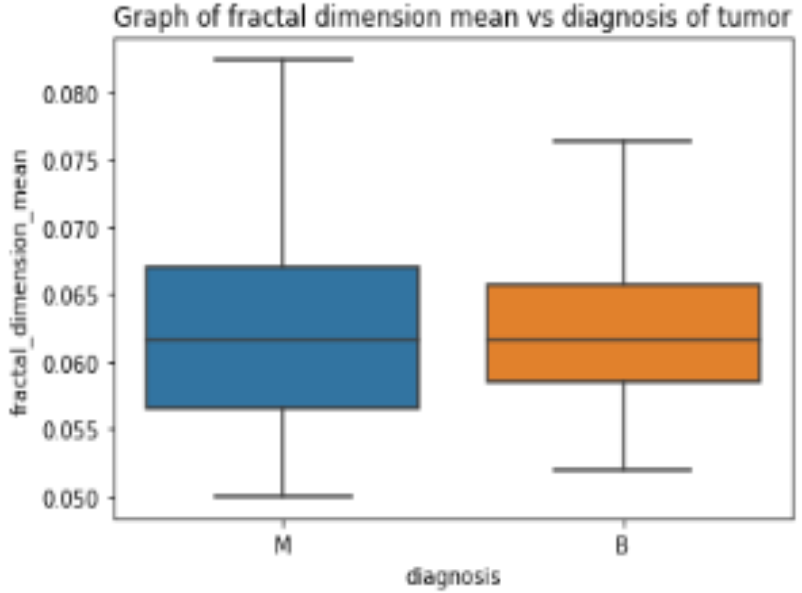
```
#Create boxplot for texture_mean vs diagnosis of tumor
plot = sns.boxplot(x='diagnosis', y='texture_mean',
data = df, showfliers= False)
plot.set_title("Graph of texture mean vs diagnosis of tumor")
```

```
Text(0.5, 1.0, 'Graph of texture mean vs diagnosis of tumor')
```




```
[ ] #Create boxplot for fractal_dimension_mean vs diagnosis of tumor
plot = sns.boxplot(x='diagnosis', y='fractal_dimension_mean',
data = df, showfliers= False)
plot.set_title("Graph of fractal dimension mean vs diagnosis of tumor")
```

```
Text(0.5, 1.0, 'Graph of fractal dimension mean vs diagnosis of tumor')
```

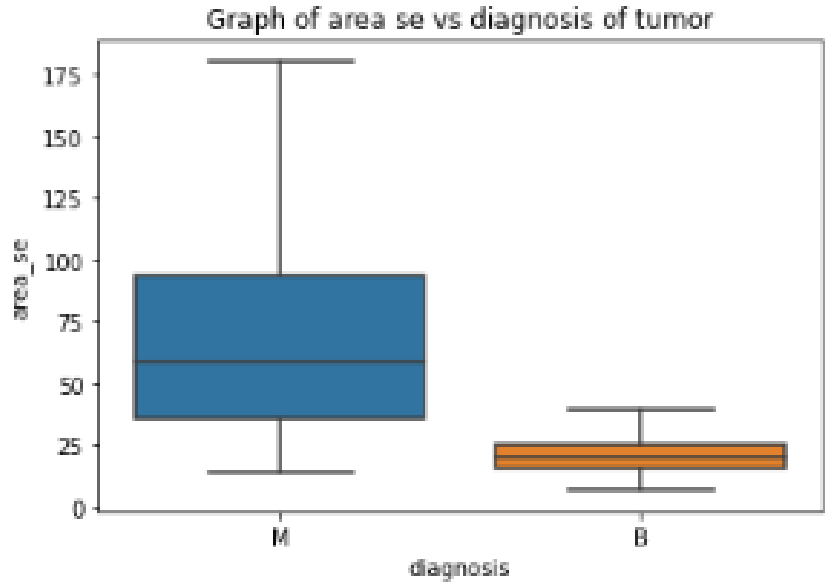


Outlier Checking

Indonesia AI
AI for Everyone, AI for Indonesia

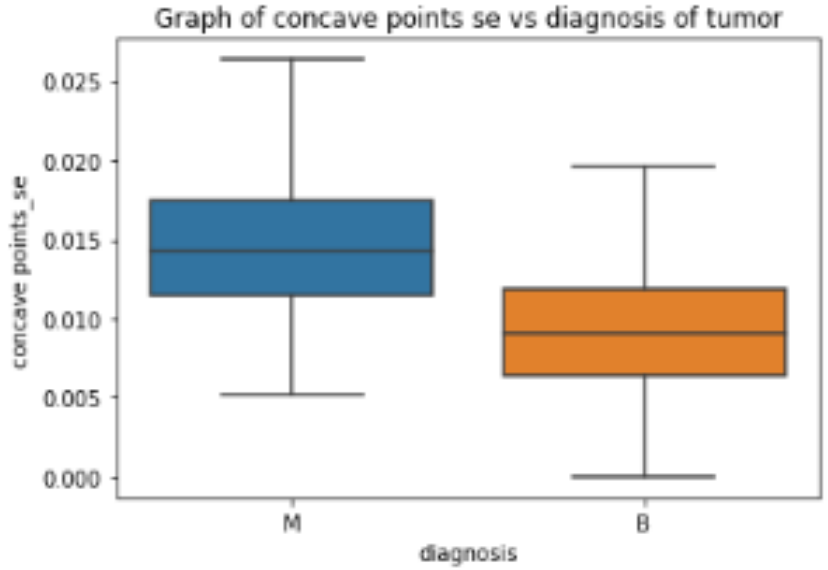
```
[ ]  #Create boxplot for area_se vs diagnosis of tumor
     plot = sns.boxplot(x='diagnosis', y='area_se',
     data = df, showfliers= False)
     plot.set_title("Graph of area se vs diagnosis of tumor")
```

```
Text(0.5, 1.0, 'Graph of area se vs diagnosis of tumor')
```




```
#Create boxplot for concave_points_se vs diagnosis of tumor
plot = sns.boxplot(x='diagnosis', y='concave points_se',
data = df, showfliers= False)
plot.set_title("Graph of concave points se vs diagnosis of tumor")
```

```
Text(0.5, 1.0, 'Graph of concave points se vs diagnosis of tumor')
```

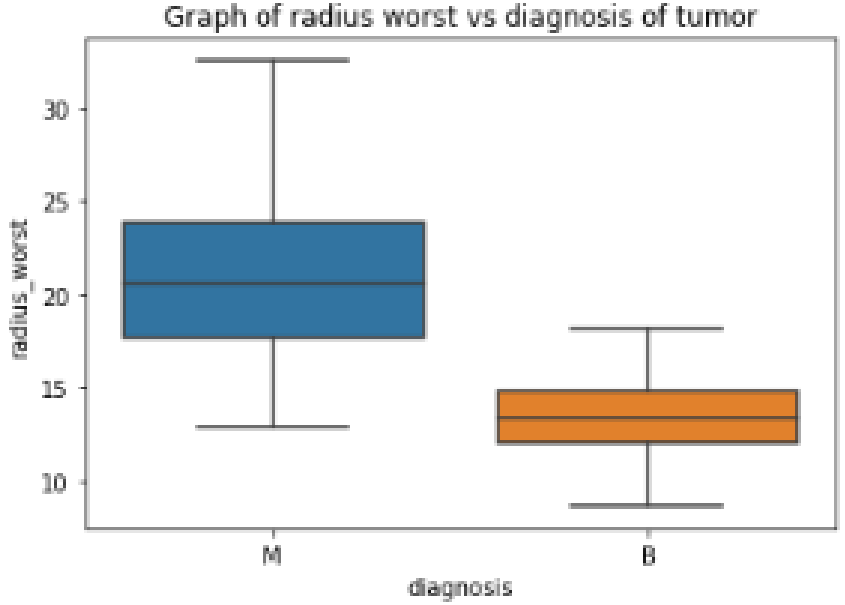


Outlier Checking

Indonesia AI
AI for Everyone, AI for Indonesia

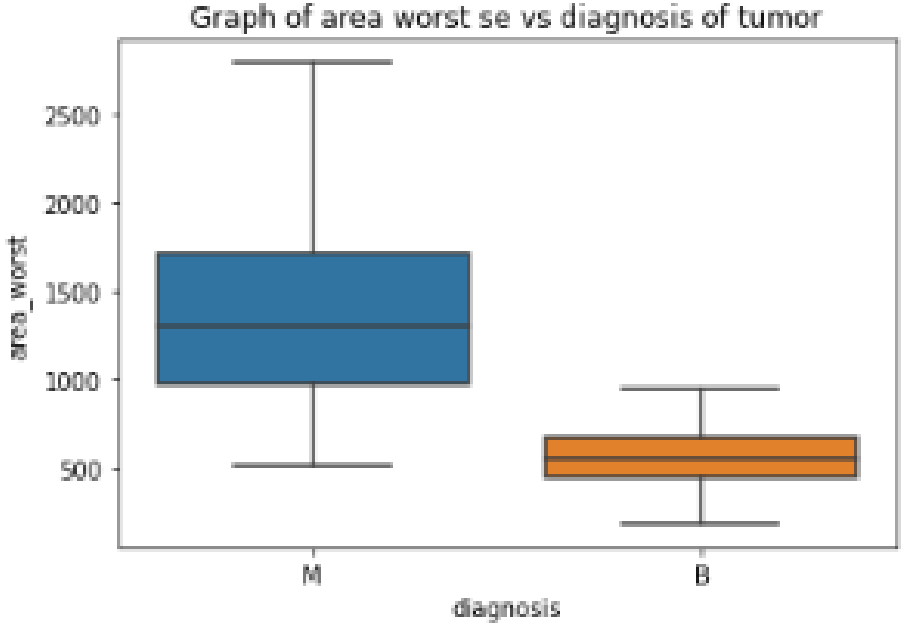
```
#Create boxplot for radius worst vs diagnosis of tumor
plot = sns.boxplot(x='diagnosis', y='radius_worst',
data = df, showfliers= False)
plot.set_title("Graph of radius worst vs diagnosis of tumor")
```

```
Text(0.5, 1.0, 'Graph of radius worst vs diagnosis of tumor')
```




```
[ ] #Create boxplot for concave_points_se vs diagnosis of tumor
plot = sns.boxplot(x='diagnosis', y='area_worst',
data = df, showfliers= False)
plot.set_title("Graph of area worst se vs diagnosis of tumor")
```

```
Text(0.5, 1.0, 'Graph of area worst se vs diagnosis of tumor')
```



Label Encoder

```
#Transform categorical value of diagnosis column using LabelEncoder
y
```

```
0      M
1      M
2      M
3      M
4      M
      ..
564    M
565    M
566    M
567    M
568    B
Name: diagnosis, Length: 569, dtype: object
```

```
[33] #Encoding categorical data values
     from sklearn.preprocessing import LabelEncoder
     labelencoder_Y = LabelEncoder()
     y= labelencoder_Y.fit_transform(y)
     print(labelencoder_Y.fit_transform(y))
```

```
[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 0 1 1 1 1 1 1 1 1 0 1 0 0 0 0 0 1 1 0 1 1 0 0 0 0 1 0 1 1 0 0 0 0 1 0 1 1
 0 1 0 1 1 0 0 0 1 1 0 1 1 1 0 0 0 1 0 0 1 1 0 0 0 1 1 0 0 0 0 1 0 0 1 0 0
 0 0 0 0 0 0 1 1 1 0 1 1 0 0 0 1 1 0 1 0 1 1 0 1 1 0 0 1 0 0 1 0 0 0 0 1 0
 0 0 0 0 0 0 0 0 1 0 0 0 0 1 1 0 1 0 0 1 1 0 0 1 1 0 0 0 0 1 0 0 1 1 1 0 1
 0 1 0 0 0 1 0 0 1 1 0 1 1 1 1 0 1 1 1 0 1 0 1 0 0 1 0 1 1 1 1 0 0 1 1 0 0
 0 1 0 0 0 0 0 1 1 0 0 1 0 0 1 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 0 0 0 0 0 0 1 0 1 0 0 1 0 0 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0
 0 1 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 1 0 0 0 0 1 1 1 0 0
 0 0 1 0 1 0 1 0 0 0 1 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1
 1 0 1 1 0 0 0 0 1 0 0 0 0 1 0 0 0 1 0 0 1 1 0 0 0 0 0 0 1 0 0 0 0 0 0
 0 1 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1 0 1 0 0 0 0 0 1 0 0
 1 0 1 0 0 1 0 1 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0
 0 0 0 0 0 0 1 0 1 0 0 1 0 0 0 0 0 0 1 1 0 1 0 1 0 0 0 0 0 1 0 0 1 0 1 0 1 1
 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 1 1 1 1 1 1 0]
```

- Sebelum data dilakukan splitting, data yang bersifat kategorik di proses dulu untuk diubah ke numerik
- Data yang bersifak kategorik adalah kolom : diagnosis
- Dilakukan label encoder dengan keteran 1: Malignant dan 0: Benign

Machine Learning

Indonesia AI
AI for Everyone, AI for Indonesia

Split Data

Indonesia AI
AI for Everyone, AI for Indonesia

```
trains_x,tests_x,trains_y,tests_y = train_test_split(x,y,test_size=0.33,random_state=42,shuffle=True, stratify=y)
```

```
print('train x_shape : ', trains_x.shape )
print('test x_shape : ', tests_x.shape )
print('train y_shape : ', trains_y.shape )
print('test y_shape : ', tests_y.shape )
```

```
train x_shape :  (381, 31)
test x_shape :  (188, 31)
train y_shape :  (381,)
test y_shape :  (188,)
```

Random Forest

Indonesia AI
AI for Everyone, AI for Indonesia

```
from sklearn.ensemble import RandomForestClassifier
```

```
classifier = RandomForestClassifier(n_estimators = 10, criterion = 'entropy', random_state = 0)
```

```
classifier.fit(trains_x, trains_y)
```

```
RandomForestClassifier(criterion='entropy', n_estimators=10, random_state=0)
```

```
Y_pred = classifier.predict(tests_x)
```

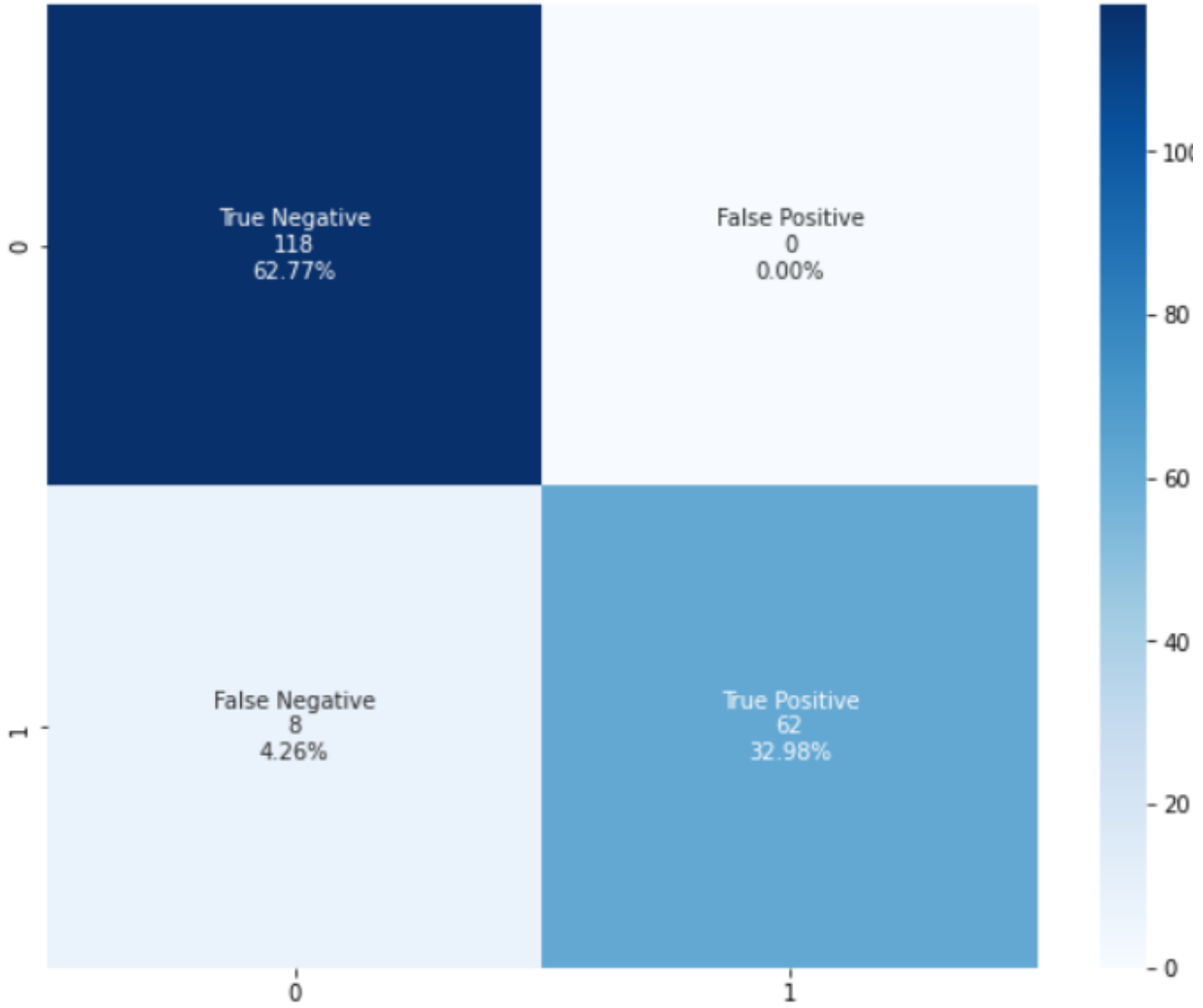
```
from sklearn.metrics import confusion_matrix
```

```
cm = confusion_matrix(tests_y, Y_pred)
```

```
group_names = ["True Negative", "False Positive", "False Negative", "True Positive"]
group_counts = ["{0:0.0f}".format(value) for value in cm.flatten()]
group_percentages = ["{0:.2%}".format(value) for value in cm.flatten()/np.sum(cm)]

labels = [f"{v1}\n{v2}\n{v3}" for v1, v2, v3 in zip(group_names,group_counts,group_percentages)]
labels = np.asarray(labels).reshape(2,2)

plt.figure(figsize=(10, 8))
sns.heatmap(cm, annot=labels, fmt='', cmap='Blues')
```

Random Forest (Result)

Indonesia AI
AI for Everyone, AI for Indonesia



Score

Indonesia AI
AI for Everyone, AI for Indonesia

Tingkat Akurasi

```
print('Model Accuracy:', score)
```

```
Model Accuracy: 0.9680851063829787
```

F1 Score

```
from sklearn.metrics import f1_score
```

```
f1_score(tests_y, Y_pred, average=None)
```

```
array([0.96721311, 0.93939394])
```

```
f1_score(tests_y, Y_pred, average='macro')
```

```
0.9533035270740189
```

Support Vector Classifier

Indonesia AI
AI for Everyone, AI for Indonesia

Support Vector Classifier

In [65]:
```
from sklearn.svm import SVC
from sklearn.model_selection import cross_val_score, KFold

from sklearn.pipeline import make_pipeline
from sklearn.metrics import confusion_matrix
from sklearn import metrics, preprocessing
from sklearn.metrics import classification_report
from sklearn.metrics import accuracy_score
```

In [66]:
```
X_train, X_test, y_train, y_test = train_test_split(Xs, y, test_size=0.33, random_state=42)

clf = SVC(probability=True)
clf.fit(X_train, y_train)

classifier_score = clf.score(X_test, y_test)
print('\nThe classifier accuracy score is {:03.2f}\n'.format(classifier_score))
```

```
The classifier accuracy score is 0.97
```

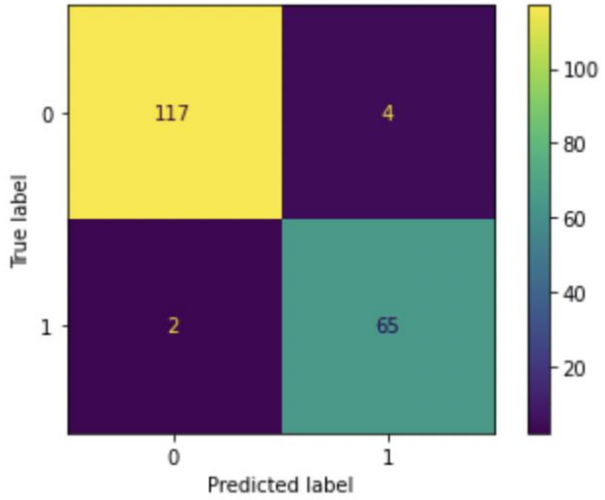
Support Vector Classifier (Result)

Indonesia AI
AI for Everyone, AI for Indonesia

In [47]:
```
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)

print(accuracy_score(y_test, y_pred))
print(confusion_matrix(y_test, y_pred))
```

```
0.9680851063829787
[[117   4]
 [  2  65]]
```

In [48]:
```
from sklearn.metrics import plot_confusion_matrix
plot_confusion_matrix(clf, X_test, y_test)
plt.show()
```



In [49]:
```
print(classification_report(y_test, y_pred ))
```

```
              precision    recall  f1-score   support

           0       0.98      0.97      0.97       121
           1       0.94      0.97      0.96        67

    accuracy                           0.97       188
   macro avg       0.96      0.97      0.97       188
weighted avg       0.97      0.97      0.97       188
```

Decision Tree Classifier

Indonesia AI
AI for Everyone, AI for Indonesia

Decision Tree Classifier

In [70]:
```
from sklearn.tree import DecisionTreeClassifier

dtc = DecisionTreeClassifier()

parameters = {
    'criterion' : ['gini', 'entropy'],
    'max_depth' : range(2, 32, 1),
    'min_samples_leaf' : range(1, 10, 1),
    'min_samples_split' : range(2, 10, 1),
    'splitter' : ['best', 'random']
}

grid_search_dt = GridSearchCV(dtc, parameters, cv = 5, n_jobs = -1, verbose = 1)
grid_search_dt.fit(X_train, y_train)
```

```
Fitting 5 folds for each of 8640 candidates, totalling 43200 fits
```

Out[70]:
```
GridSearchCV(cv=5, estimator=DecisionTreeClassifier(), n_jobs=-1,
             param_grid={'criterion': ['gini', 'entropy'],
                         'max_depth': range(2, 32),
                         'min_samples_leaf': range(1, 10),
                         'min_samples_split': range(2, 10),
                         'splitter': ['best', 'random']},
             verbose=1)
```

In [71]:
```
grid_search_dt.best_params_
```

Out[71]:
```
{'criterion': 'entropy',
 'max_depth': 25,
 'min_samples_leaf': 4,
 'min_samples_split': 2,
 'splitter': 'random'}
```

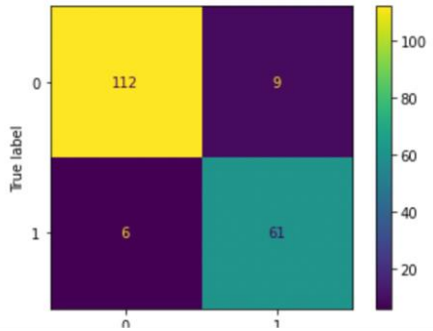
In [72]:
```
grid_search_dt.best_score_
```

Out[72]:
```
0.9606288448393713
```

Decision Tree Classifier (Result)

Indonesia AI
AI for Everyone, AI for Indonesia

```
In [73]: dtc = DecisionTreeClassifier(criterion = 'entropy', max_depth = 28, min_samples_leaf = 1, min_samples_split = 8, spl
         dtc.fit(X_train, y_train)

Out[73]: DecisionTreeClassifier(criterion='entropy', max_depth=28, min_samples_split=8,
                                splitter='random')

In [74]: y_pred = dtc.predict(X_test)

In [75]: print(accuracy_score(y_train, dtc.predict(X_train)))

         dtc_acc = accuracy_score(y_test, dtc.predict(X_test))
         print(dtc_acc)

         0.9763779527559056
         0.9202127659574468

In [78]: print(confusion_matrix(y_test, y_pred))
         from sklearn.metrics import plot_confusion_matrix
         plot_confusion_matrix(dtc, X_test, y_test)
         plt.show()
         print(classification_report(y_test, y_pred))

         [[112   9]
          [  6  61]]
```




```
              precision    recall  f1-score   support

           0       0.95      0.93      0.94       121
           1       0.87      0.91      0.89        67

    accuracy                           0.92       188
   macro avg       0.91      0.92      0.91       188
weighted avg       0.92      0.92      0.92       188
```

Conclusion

- Data sudah clean dibuktikan dengan tidak ada duplicate values, missing values dan nilai null. Serta sudah dibedakan untuk fitur mean, standard error dan worst
- Confusion matrix yang dipilih adalah F1 score karena F1 score dapat menangani data imbalance dan meminimalkan false positive dan false negative
- Berdasarkan beberapa algoritma yang telah dicoba(Random Forest, Decision Tree, SVC) nilai F1 score tertinggi terdapat pada SVC dengan nilai 0.97 tidak berbeda jauh dengan Random Forest yaitu sebesar 0.95.

Reccomendation

- Perlu dilakukan hyperparameter tuning untuk memastikan model tidak overfitting
- Pemilihan fitur tertentu untuk mencegah multi kolienaritas

Indonesia AI
AI for Everyone, AI for Indonesia

Terima Kasih!