LAPORAN TUGAS BESAR

PEMBELAJARAN MESIN LANJUT (CII3L3)

KELAS MATA KULIAH PEMBELAJARAN MESIN IF-PIL-IS01 (SUO)

Adhitya Melani Eka Janarwati (1301204046)

Muhammad Daffa' Ibrahim (1301204051)

Program Studi Sarjana Informatika

Fakultas Informatika

Universitas Telkom

Bandung

2023

DAFTAR ISI

BAB I PENDAHULUAN

1. Latar Belakang

Machine Learning sudah menjadi sebuah teknologi yang cukup populer dan banyak digunakan oleh industri. Sistem ini dapat mempelajari hal baru berdasarkan hasil pengolahan dari data yang tersedia. Perkembangan machine learning dalam dunia industri sangatlah cepat dan terus berkembang hingga kini. Terciptanya metode-metode machine learning yang lebih optimal dan akurat serta mudah untuk diimplementasikan seperti *Deep Learning* dan salah satunya adalah *Automated Machine Learning.*

Automated Machine Learning biasa disebut dengan istilah AutoML merupakan salah satu subbidang yang sedang populer dalam ilmu data. AutoML memungkinkan mesin untuk mempelajari kecerdasan buatan sendiri secara mandiri. Proses pengolahan data menggunakan AutoML dapat menghilangkan kebutuhan akan *data scientist*. Hal ini dikarenakan AutoML dapat memproses *data cleaning, feature selection, model selection*, serta *parameter selection* secara otomatis.

2. Pembahasan Dataset

Dataset yang digunakan adalah dataset “Breast Cancer Wisconsin”. Dataset ini merupakan sekumpulan data mengenai penyakit kanker payudara. Pada dataset terdapat beberapa informasi mengenai kanker payudara seperti bentuk, ukuran, tekstur, dan informasi lainnya, serta terdapat informasi jenis kanker payudara yaitu jinak dan ganas. Project ini memiliki tujuan yaitu membuat model untuk dapat memperkirakan jenis kanker dalam hal ini (jinak/ganas) berdasarkan informasi atau fitur-fitur yang dimiliki pada dataset “Breast Cancer Wisconsin”. Dalam project ini, pemodelan akan dilakukan menggunakan metode Automated Machine Learning dengan tools yang digunakan merupakan tools dari python yaitu TPOT.

```
#Membaca data train
df = pd.read_csv("train.csv")
df
```

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	...	radius_worst	texture_worst
0	842302	M	17.990	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.1471	...	25.38	17.33
1	842517	M	20.570	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	...	24.99	23.41
2	84300903	M	19.690	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.1279	...	23.57	25.53
3	84348301	M	11.420	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.1052	...	14.91	26.50
4	84358402	M	20.290	14.34	135.10	1297.0	0.10030	0.13280	0.198	0.1043	...	22.54	16.67
...	...	...	...	...	...	...	...	...	...	...	...	...	...
458	9112594	B	13.000	25.13	82.61	520.2	0.08369	0.05073	0.01206	0.01762	...	14.34	31.88
459	9112712	B	9.755	28.20	61.68	290.9	0.07984	0.04626	0.01541	0.01043	...	10.67	36.92
460	911296201	M	17.080	27.15	111.20	930.9	0.09898	0.11100	0.1007	0.06431	...	22.96	34.49
461	911296202	M	27.420	26.27	186.90	2501.0	0.10840	0.19880	0.3635	0.1689	...	36.04	31.37
462	9113156	B	14.400	26.99	92.25	646.1	0.06995	0.05223	0.03476	0.01737	...	15.40	31.98

463 rows × 32 columns

Pada dataset “Breast Cancer Wisconsin” dapat dilihat terdapat 463 baris record data serta terdapat 32 kolom fitur. Atribut “diagnosis” merupakan atribut target yang memberikan informasi jenis kanker payudara yaitu data dengan label M yang berarti *Malignant* atau ganas serta data dengan label B yang berarti *Beningn* atau jinak.

BAB II Data Preparation and Data Pre-Processing

Sebelum melakukan pengolahan terhadap data atau data modelling. Perlu adanya data preparation serta data pre-processing agar dataset terhindar dari noise serta memudahkan proses pemodelan yang dilakukan. Selain itu juga, data perlu dibersihkan dan dirapikan agar hasil pemodelan dapat optimal.

1. Memahami Dataset

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 463 entries, 0 to 462
Data columns (total 32 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   id                      463 non-null    int64
 1   diagnosis               463 non-null    object
 2   radius_mean             463 non-null    float64
 3   texture_mean            463 non-null    float64
 4   perimeter_mean          463 non-null    float64
 5   area_mean               463 non-null    float64
 6   smoothness_mean         463 non-null    float64
 7   compactness_mean        463 non-null    float64
 8   concavity_mean          463 non-null    object
 9   concave points_mean     463 non-null    object
 10  symmetry_mean           463 non-null    float64
 11  fractal_dimension_mean  463 non-null    float64
 12  radius_se               463 non-null    float64
 13  texture_se              463 non-null    float64
 14  perimeter_se            463 non-null    float64
 15  area_se                 463 non-null    float64
 16  smoothness_se           463 non-null    float64
 17  compactness_se          463 non-null    float64
 18  concavity_se            463 non-null    object
 19  concave points_se       463 non-null    object
 20  symmetry_se             463 non-null    float64
 21  fractal_dimension_se    463 non-null    float64
 22  radius_worst            463 non-null    float64
 23  texture_worst           463 non-null    float64
 24  perimeter_worst         463 non-null    float64
 25  area_worst              463 non-null    float64
 26  smoothness_worst        463 non-null    float64
 27  compactness_worst       463 non-null    float64
 28  concavity_worst         463 non-null    object
 29  concave points_worst    463 non-null    object
 30  symmetry_worst          463 non-null    float64
 31  fractal_dimension_worst  463 non-null    float64
dtypes: float64(24), int64(1), object(7)
memory usage: 115.9+ KB
```

Berdasarkan informasi pada gambar diatas, dapat diamati bahwa atribut yang terdapat pada dataset memiliki tipe yang beragam yaitu integer, float, serta object.

```
#check volume data train
df.shape
```

```
(463, 32)
```

Selanjutnya melakukan pengamatan terhadap volume dataset, terdapat 463 baris record serta 32 kolom atribut pada dataset tersebut.

2. Penanggulangan missing value

```
#checking missing value for each feature
print('Checking missing value for each feature:')
print(df.isnull().sum())
#Counting total missing value
print('\nCounting total missing value:')
print(df.isnull().sum().sum())
```

```
Checking missing value for each feature:
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

Counting total missing value:
0
```

Dapat diamati, berdasarkan hasil pengecekkan diatas bahwa dataset tidak memiliki missing value. Sehingga tidak perlu dilakukan penanggulangan terhadap record data yang memiliki missing value.

3. Mengganti tipe data atribut target menjadi numerik

```
#diagnosis sebagai target variable
df['diagnosis'] = df['diagnosis'].replace({'B':0,'M':1})
print("df_train diagnosis:",df['diagnosis'].unique())
```

```
df_train diagnosis: [1 0]
```

Berdasarkan informasi yang didapatkan sebelumnya, atribut target yaitu "diagnosis" merupakan atribut dengan tipe data object. Langkah yang kami ambil adalah mengubah tipe data atribut "diagnosis" menjadi numerik, hal ini dilakukan untuk mempermudah proses pemodelan serta pengolahan data.

4. Mengganti atribut bertipe data object menjadi numerik

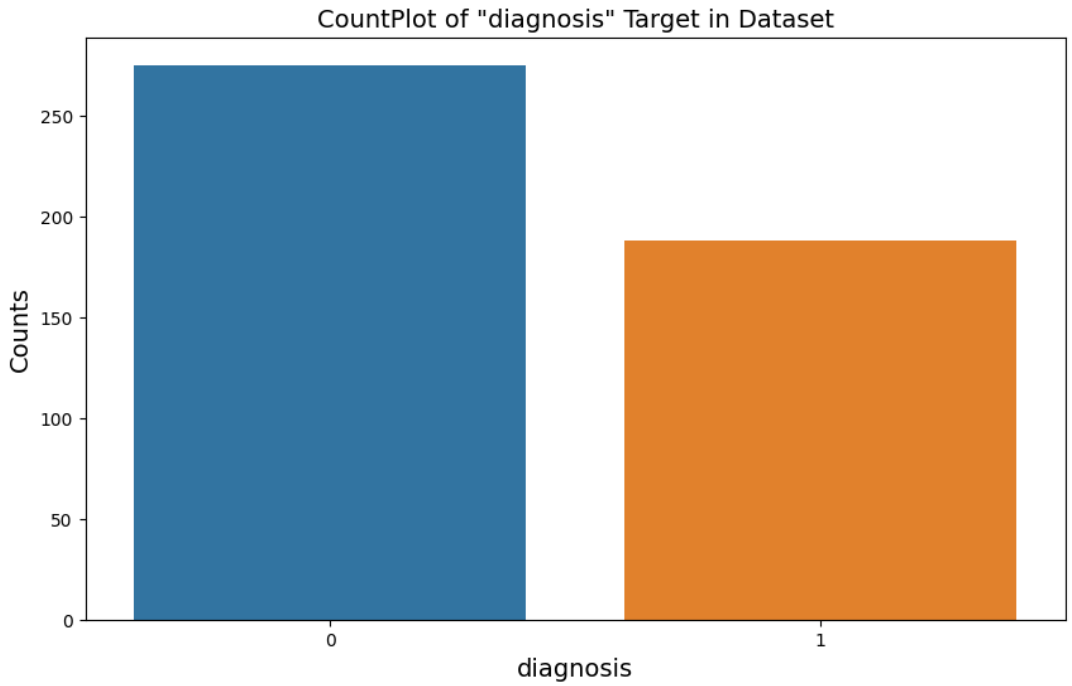
```
numerical_features = df.select_dtypes(include=np.number).columns.tolist()
categorical_features = df.select_dtypes(include='object').columns.tolist()

print('Numerical Features:', numerical_features)
print('\nCategorigal Features:', categorical_features)
```

```
Numerical Features: ['id', 'diagnosis', 'radius_mean', 'texture_mean', 'perimeter_

Categorigal Features: ['concavity_mean', 'concave points_mean', 'concavity_se', '
```

```
for i in categorical_features:
  df[i] = pd.to_numeric(df[i], errors='coerce')
  df[i] = pd.to_numeric(df[i], errors='coerce')
```

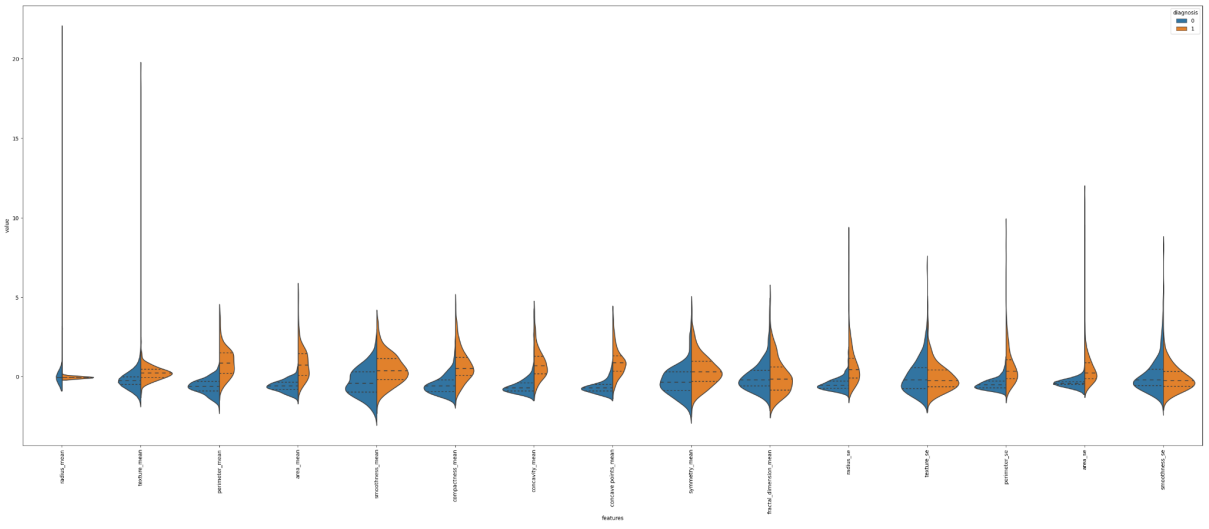
Berdasarkan informasi yang diperoleh sebelumnya, terdapat beberapa atribut dengan tipe data object, hal ini tentu akan membuat proses pengolahan data menjadi rumit dan kompleks. Langkah yang kami ambil adalah mengubah tipe data menjadi numerik pada atribut yang memiliki tipe data object. Dengan cara membagi kelompok atribut dengan tipe data object, kemudian diubah menjadi tipe data numerik.

5. Visualisasi Atribut “Diagnosis”



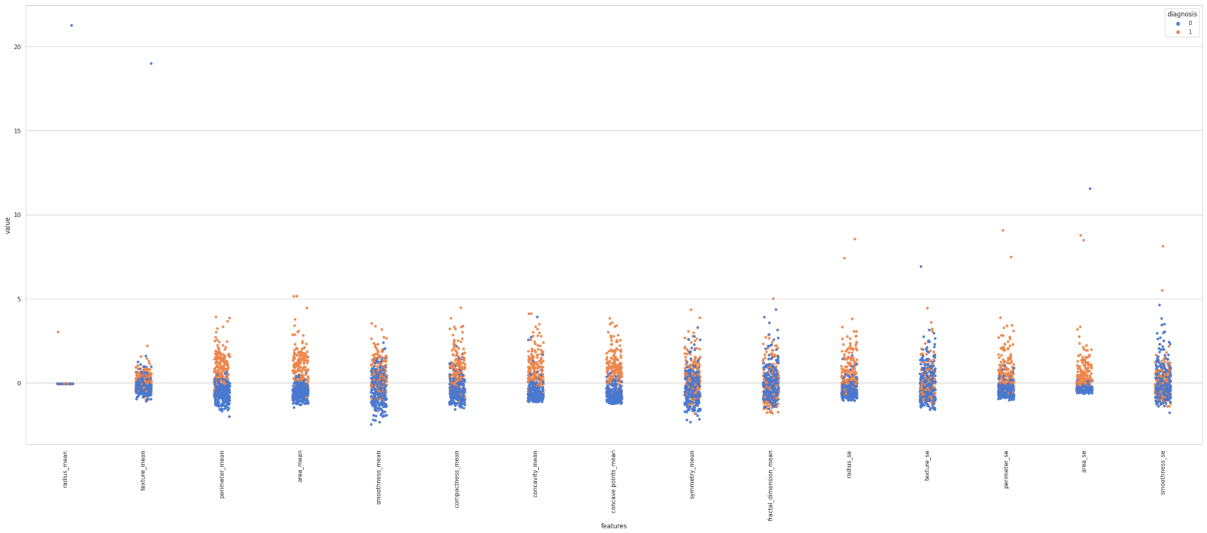
Melihat perbandingan data pada atribut “diagnosis”. Berdasarkan visualisasi data diatas, terlihat bahwa atribut “diagnosis” yang bernilai 0 atau ganas memiliki jumlah data yang lebih banyak dibandingkan dengan jumlah data dengan atribut “diagnosis” bernilai 1. Hal ini menandakan bahwa kanker ganas lebih banyak dibandingkan dengan kanker jinak pada datasets tersebut.

6. Visualisasi Menggunakan Violin Plot



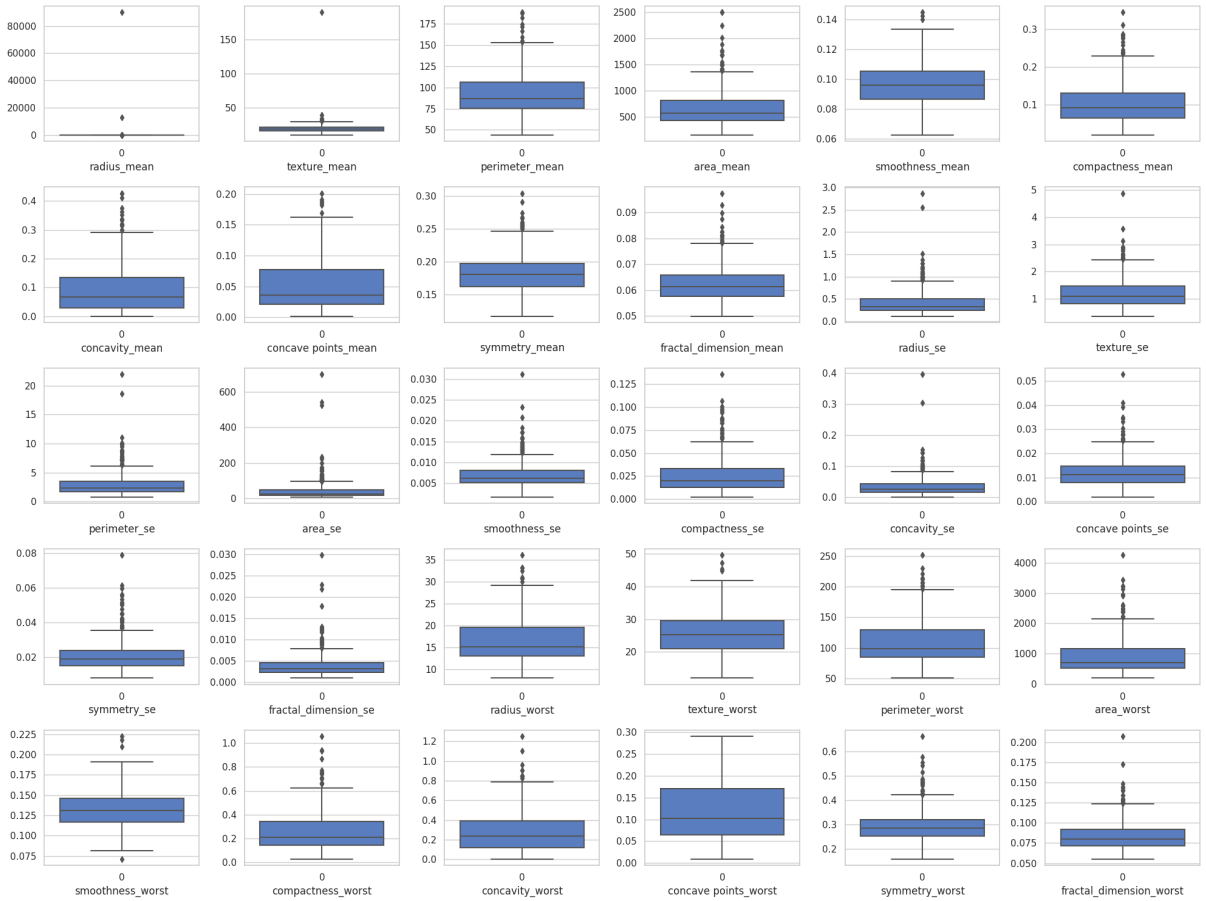
Violin Plot berguna untuk mengetahui distribusi dari continous variabel yang ada dan juga menunjukan density dari distribusinya. ketebalan plot menunjukan density atau kepadatan datanya yang mana semakin besar plotnya menunjukan makin banyak data yang berkumpul pada nilai tersebut. Garis horizontal dalam plot mewakili nilai median data, dan kotak menunjukkan rentang interkuartil (IQR), yang berisi 50% data tengah. Berikut Violin Plot pada dataset.

7. Visualisasi menggunakan Stir Plot



Stir Plot merupakan plot sederhana plot sederhana di mana setiap titik data diwakili oleh titik di sepanjang sumbu variabel yang diukur. Dengan warna yang ada, kita bisa membandingkan distribusi variabel yang sama di seluruh grup atau kategori yang berbeda. Terlihat ada beberapa pencilan yang sangat ekstrim diantara beberapa nilai yang sangat kecil, contohnya pada radius mean, texture mean, concavity se, serta fractal dimension se.

8. Visualisasi menggunakan Box Plot



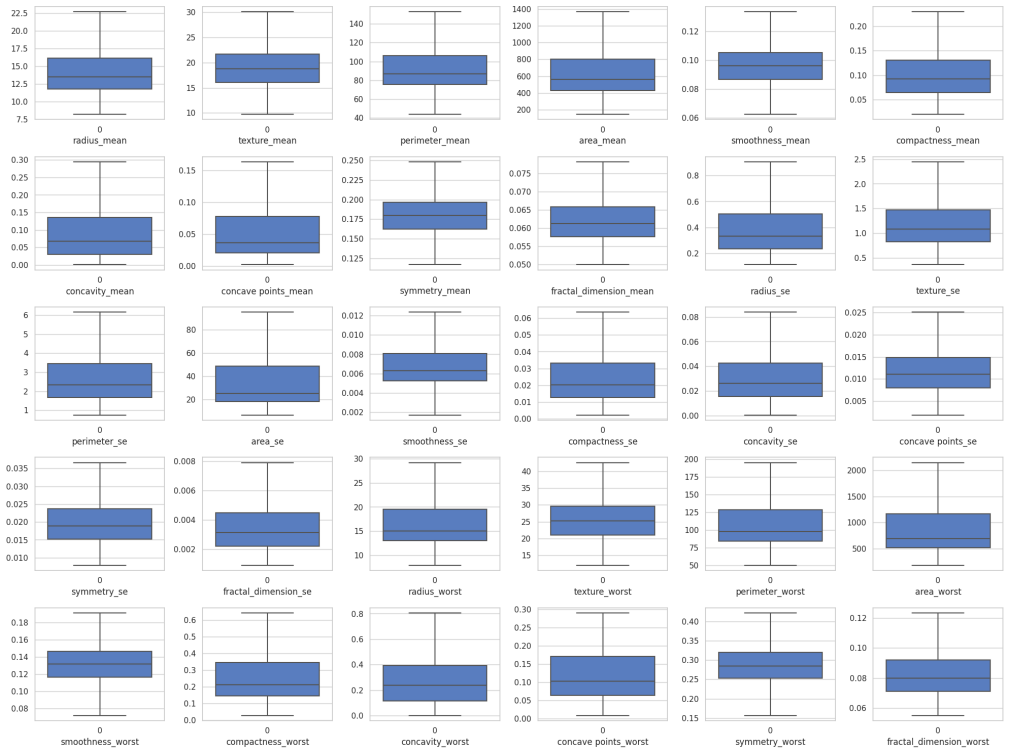
Boxplot digunakan untuk visualisasi data yang menampilkan distribusi kontinu. Dalam Box Plot, persegi panjang digambar untuk mewakili rentang interkuartil (IQR), yang berisi 50% data tengah. Garis vertikal pada gambar di dalam persegi panjang untuk mewakili nilai median data. Whiskers digambar dari atas dan bawah persegi panjang ke pengamatan terbesar dan terkecil yang masih dalam 1,5 kali IQR median. Berdasarkan Boxplot di atas, terdapat banyak pencilan data yang dapat mempengaruhi performa data pada saat diolah.

9. Handling Outlier

```
# handle outlier using interquartile range (IQR) method
def mod_outlier(df):
        df1 = df.copy()
        df = df._get_numeric_data()


        q1 = df.quantile(0.25)
        q3 = df.quantile(0.75)

        iqr = q3 - q1

        lower_bound = q1 -(1.5 * iqr)
        upper_bound = q3 +(1.5 * iqr)


        for col in df.columns:
            for i in range(0,len(df[col])):
                if df.loc[i, col] < lower_bound[col]:
                    df.loc[i, col] = lower_bound[col]

                if df.loc[i, col] > upper_bound[col]:
                    df.loc[i, col] = upper_bound[col]


        for col in df.columns:
            df1[col] = df[col]

        return(df1)

df = mod_outlier(df)
```

Terdapat beberapa outlier data yang dapat mempengaruhi performa data pada saat data diolah. Outlier tersebut ditangani dengan memberi batasan terhadap outlier. Batas nilai maksimum yang digunakan adalah nilai (q3 + (interquartile * 1.5)) sedangkan batas nilai minimum yang digunakan adalah nilai (q1 - (interquartile * 1.5)). Jika terdapat nilai yang melebihi batas maksimum, maka nilai tersebut akan di assign dengan nilai pada batas maksimum, Jika terdapat nilai yang lebih kecil batas minimum, maka nilai tersebut akan di assign dengan nilai pada batas minimum.

Berikut merupakan visualisasi box plot setelah dilakukan handling outlier



10. Visualisasi data menggunakan Diagram Batang

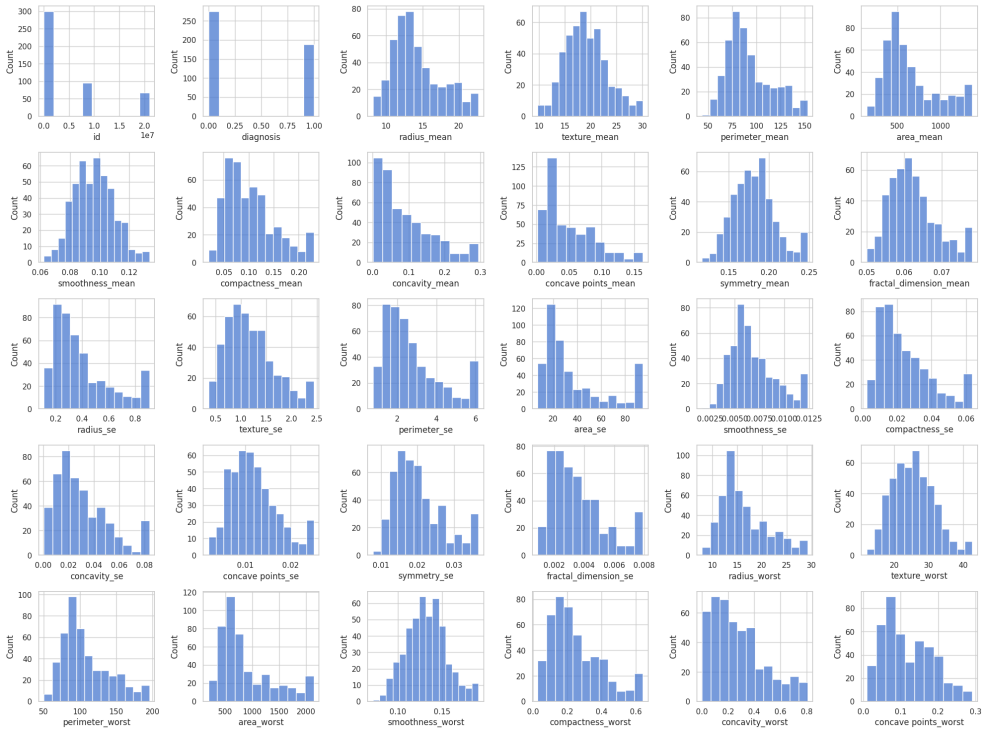
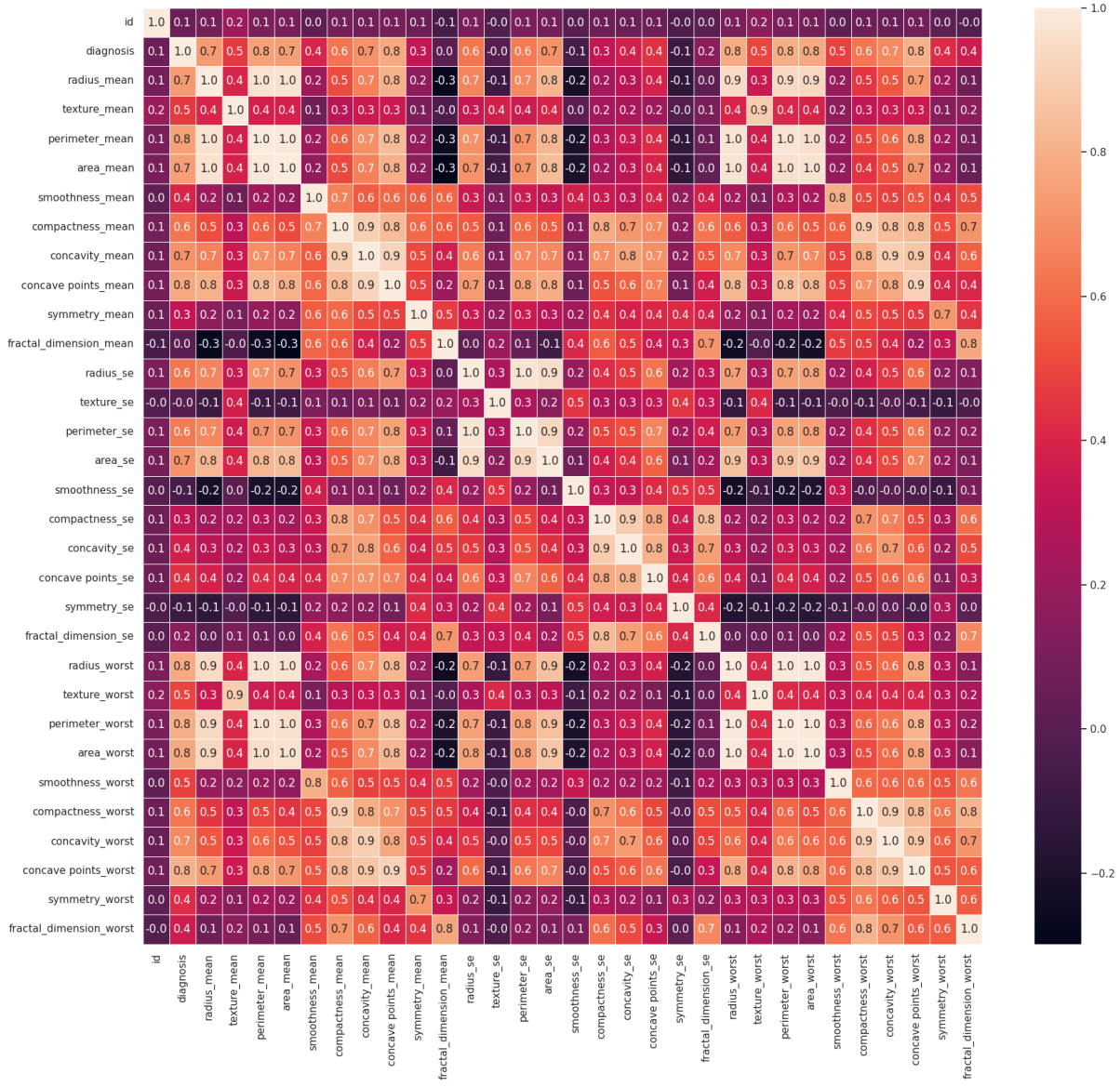


Diagram Batang digunakan untuk melakukan visualisasi data yang menampilkan jumlah data berdasarkan value atau nilai dari data tersebut yang dikelompokkan berdasarkan atribut dari datasets.

11. Visualisasi data menggunakan Heat map



Berdasarkan visualisasi heatmap diatas, dapat diamati nilai korelasi setiap atribut terutama nilai korelasi terhadap atribut “diagnosis”. Nilai korelasi merupakan nilai hubungan antar atribut, terdapat nilai korelasi positif ketika kedua atribut memiliki arah yang sama, sedangkan nilai korelasi negatif ketika kedua atribut memiliki arah yang berlawanan.

12. Reduksi Dimensi

```
columns = np.full((corr.shape[0],), True, dtype=bool)
for i in range(corr.shape[0]):
    for j in range(i+1, corr.shape[0]):
      if corr.iloc[i,j] >= 0.95:
        if columns[j]:
          columns[j] = False

selected_columns = df.columns[columns]
print("sum of selected column:", selected_columns.shape[0])
print("selected column:", selected_columns)
```

```
sum of selected column: 26
selected column: Index(['id', 'diagnosis', 'radius_mean', 'texture_mean', 'smoothness_mean',
       'compactness_mean', 'concavity_mean', 'concave points_mean',
       'symmetry_mean', 'fractal_dimension_mean', 'radius_se', 'texture_se',
       'area_se', 'smoothness_se', 'compactness_se', 'concavity_se',
       'concave points_se', 'symmetry_se', 'fractal_dimension_se',
       'texture_worst', 'smoothness_worst', 'compactness_worst',
       'concavity_worst', 'concave points_worst', 'symmetry_worst',
       'fractal_dimension_worst'],
      dtype='object')
```

Berdasarkan visualisasi heatmap di atas, terdapat beberapa atribut yang memiliki nilai korelasi rendah serta memiliki kemiripan terhadap atribut lain. Sehingga diperlukannya proses reduksi dimensi atau feature selection untuk meningkatkan performa data pada saat dilakukan pemodelan. Berdasarkan gambar diatas, dapat dilihat bahwa terdapat 26 atribut yang dipilih untuk kemudian diolah serta dilakukan pemodelan.

13. Splitting Data

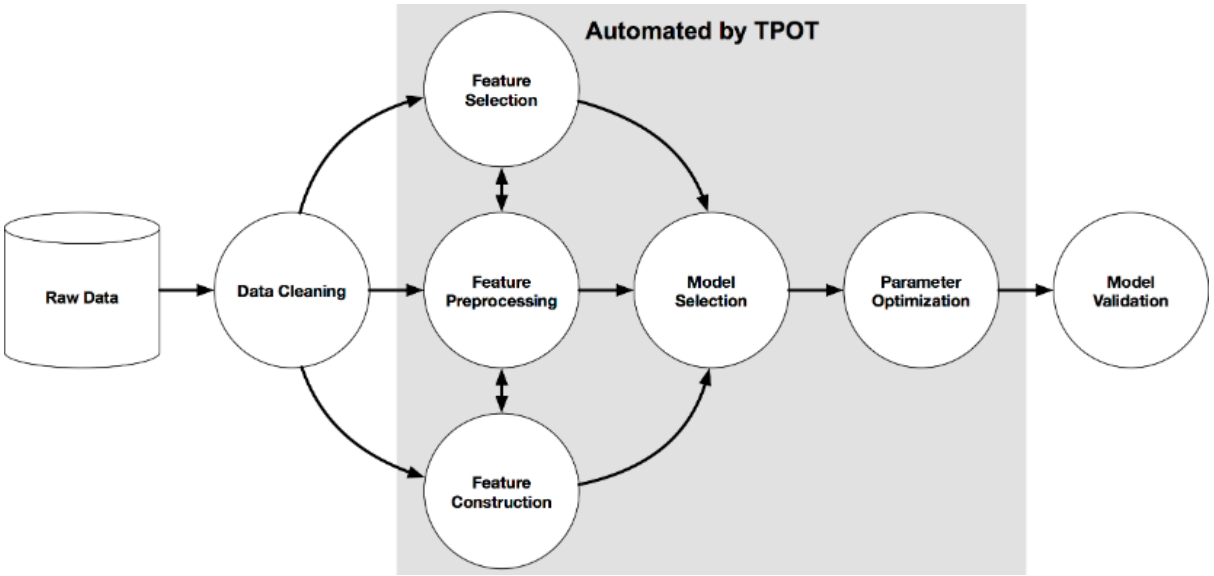
```
y = df["diagnosis"]
X = df.drop(["id", "diagnosis"], axis = 1)

X_train,X_test,y_train,y_test = train_test_split(X,y,test_size = 0.2, random_state = 42)
```

Langkah terakhir dalam data preparing dan data pre-processing adalah melakukan splitting data. Data dibagi menjadi data latih dan data uji dengan perbandingan 80% data latih dan 20% data uji serta random_state di set dengan 42.

BAB III PEMODELAN

1. Automated Machine Learning Tools

Automated Machine Learning Tools yang digunakan pada project ini adalah TPOT. TPOT (Tree-based Pipeline Optimization Tool) merupakan library open source untuk menggunakan Automated machine learning dengan python. TPOT merupakan tools yang secara otomatis membuat dan mengoptimalkan pipeline model pada machine learning menggunakan pemrograman genetik. TPOT akan secara otomatis melakukan eksplorasi terhadap berbagai pipeline model dan menentukan model terbaik untuk datasets yang digunakan.



Berikut merupakan gambaran singkat mengenai cara kerja TPOT sebagai Automated Machine Learning. Berdasarkan graph diatas, dapat dilihat bahwa TPOT akan secara otomatis melakukan beberapa tahap dalam pengolahan datasets, diantaranya adalah *Feature Selection, Feature Preprocessing, Feature Construction, Model Selection,* Serta *Parameter Optimization.* Dalam project ini, TPOT akan secara otomatis melakukan proses pemodelan hingga menemukan model terbaik berdasarkan dari data serta atribut-atribut yang terdapat pada datasets “Breast Cancer Wisconsin”.

2. Pemodelan Menggunakan TPOT

```
model = tpot.TPOTClassifier(generations=8, population_size=50, verbosity=2)
model.fit(X_train, y_train)
```

```
Imputing missing values in feature set


Generation 1 - Current best internal CV score: 0.972972972972973

Generation 2 - Current best internal CV score: 0.9756756756756756

Generation 3 - Current best internal CV score: 0.9756756756756756

Generation 4 - Current best internal CV score: 0.9756756756756756

Generation 5 - Current best internal CV score: 0.9756756756756758

Generation 6 - Current best internal CV score: 0.9756756756756758

Generation 7 - Current best internal CV score: 0.9783783783783784

Generation 8 - Current best internal CV score: 0.9783783783783784

Best pipeline: GradientBoostingClassifier(PolynomialFeatures(input_matrix, degree=2, include_bias=False,
```

```
TPOTClassifier
TPOTClassifier(generations=8, population_size=50, verbosity=2)
```

Setelah melakukan proses preparing data dan pre-processing data, langkah selanjutnya adalah melakukan pemodelan menggunakan TPOT. Pada pemodelan ini kami mengatur parameter dengan generations=8, jumlah populasi= 3 dan verbosity=2. Dari model tersebut, dihasilkan CV Score terbesar yaitu 0.978.

BAB IV EVALUASI

1. Akurasi Model

```
#Menlihat akurasi
print("Accuracy is {}%".format(model.score(X_test, y_test)*100))
```

```
Imputing missing values in feature set
Accuracy is 97.84946236559139%
```

Setelah melakukan pemodelan menggunakan data latih, selanjutnya menghitung nilai akurasi menggunakan data uji. Berdasarkan pemodelan yang dilakukan, dihasilkan akurasi terhadap data uji sebesar 97.8%

2. Best Pipeline

```
#Best pipeline
print("Best pipeline:", model.fitted_pipeline_)
```

```
Best pipeline: Pipeline(steps=[('polynomialfeatures', PolynomialFeatures(include_bias=False)),
                ('gradientboostingclassifier',
                 GradientBoostingClassifier(max_depth=2, max_features=0.45,
                                            min_samples_leaf=3,
                                            min_samples_split=16,
                                            subsample=0.4))])
```

TPOT dapat mengotomatisasi pemilihan model terbaik berdasarkan datasets yang digunakan. Setelah melakukan pemodelan menggunakan datasets “Breast Cancer Wisconsin”, didapatkan best pipeline menggunakan GradientBoostingClassifier bersamaan dengan parameter terbaik sesuai yang ada pada gambar hasil pemodelan diatas.

3. Eksperimen

```
eksperimen1 = tpot.TPOTClassifier(generations=5, population_size=50, verbosity=2)
eksperimen1.fit(X_train, y_train)
```

```
Imputing missing values in feature set

Generation 1 - Current best internal CV score: 0.9675675675675676

Generation 2 - Current best internal CV score: 0.9675675675675677

Generation 3 - Current best internal CV score: 0.9675675675675677

Generation 4 - Current best internal CV score: 0.9783783783783784

Generation 5 - Current best internal CV score: 0.9783783783783784

Best pipeline: LinearSVC(MLPClassifier(MaxAbsScaler(input_matrix), alpha=0.01, learning_rate_ini
```

```
TPOTClassifier
TPOTClassifier(generations=5, population_size=50, verbosity=2)
```

```
[ ] #Menlihat akurasi
    print("Accuracy is {}%".format(eksperimen1.score(X_test, y_test)*100))
```

```
Imputing missing values in feature set
Accuracy is 95.6989247311828%
```

Eksperimen pertama dengan mengatur parameter generation=5 dan population size=50. Didapatkan hasil best CV score sebesar 0.97 serta akurasi pada data test sebesar 95.6%

```
eksperimen2 = tpot.TPOTClassifier(generations=5, population_size=100, verbosity=2)
eksperimen2.fit(X_train, y_train)
```

```
Imputing missing values in feature set

Generation 1 - Current best internal CV score: 0.9756756756756758

Generation 2 - Current best internal CV score: 0.9756756756756758

Generation 3 - Current best internal CV score: 0.9810810810810813

Generation 4 - Current best internal CV score: 0.9810810810810813

Generation 5 - Current best internal CV score: 0.9837837837837837

Best pipeline: MLPClassifier(MaxAbsScaler(StandardScaler(input_matrix)), alpha=0.1,
```

```
TPOTClassifier
TPOTClassifier(generations=5, verbosity=2)
```

```
#Menlihat akurasi
print("Accuracy is {}%".format(eksperimen2.score(X_test, y_test)*100))
```

```
Imputing missing values in feature set
Accuracy is 96.7741935483871%
```

Eksperimen kedua dengan mengatur parameter generation=5 dan population size=100. Didapatkan hasil best CV score sebesar 0.983 serta akurasi pada data test sebesar 96.7%

```
eksperimen3 = tpot.TPOTClassifier(generations=5, population_size=150, verbosity=2)
eksperimen3.fit(X_train, y_train)
```

```
Imputing missing values in feature set

Generation 1 - Current best internal CV score: 0.9783783783783784

Generation 2 - Current best internal CV score: 0.9783783783783784

Generation 3 - Current best internal CV score: 0.9783783783783784

Generation 4 - Current best internal CV score: 0.9783783783783784

Generation 5 - Current best internal CV score: 0.9783783783783784

Best pipeline: LinearSVC(MinMaxScaler(input_matrix), C=5.0, dual=False, loss=squared
```

TPOTClassifier

```
TPOTClassifier(generations=5, population_size=150, verbosity=2)
```

```
#Menlihat akurasi
print("Accuracy is {}%".format(eksperimen3.score(X_test, y_test)*100))
```

```
Imputing missing values in feature set
Accuracy is 95.6989247311828%
```

Eksperimen ketiga dengan mengatur parameter generation=5 dan population size=150. Didapatkan hasil best CV score sebesar 0.978 serta akurasi pada data test sebesar 95.6%

```
eksperimen4 = tpot.TPOTClassifier(generations=10, population_size=50, verbosity=2)
eksperimen4.fit(X_train, y_train)
```

```
Imputing missing values in feature set

Generation 1 - Current best internal CV score: 0.9756756756756758

Generation 2 - Current best internal CV score: 0.981081081081081

Generation 3 - Current best internal CV score: 0.981081081081081

Generation 4 - Current best internal CV score: 0.981081081081081

Generation 5 - Current best internal CV score: 0.981081081081081

Generation 6 - Current best internal CV score: 0.981081081081081

Generation 7 - Current best internal CV score: 0.981081081081081

Generation 8 - Current best internal CV score: 0.981081081081081

Generation 9 - Current best internal CV score: 0.981081081081081

Generation 10 - Current best internal CV score: 0.981081081081081

Best pipeline: MLPClassifier(BernoulliNB(MinMaxScaler(input_matrix), alpha=0.001, fit_
```

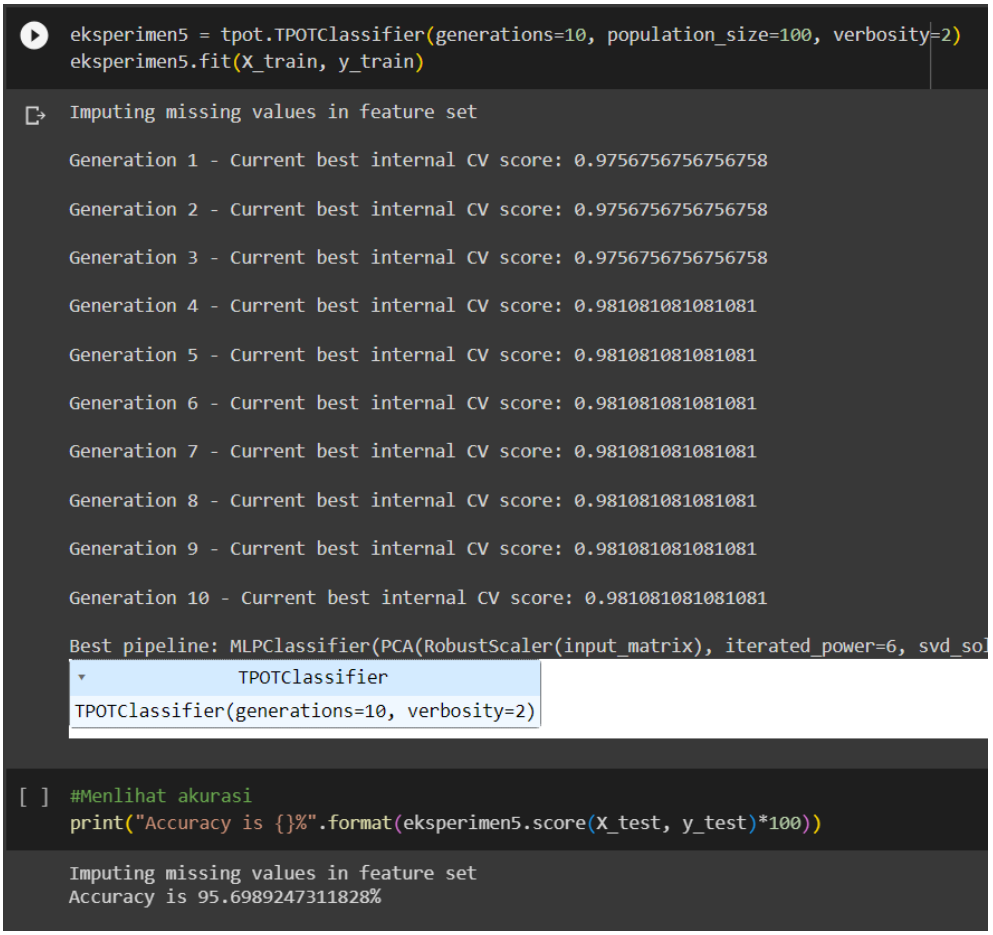
TPOTClassifier

```
TPOTClassifier(generations=10, population_size=50, verbosity=2)
```

```
#Menlihat akurasi
print("Accuracy is {}%".format(eksperimen4.score(X_test, y_test)*100))
```

```
Imputing missing values in feature set
Accuracy is 96.7741935483871%
```

Eksperimen keempat dengan mengatur parameter generation=10 dan population size=50. Didapatkan hasil best CV score sebesar 0.981 serta akurasi pada data test sebesar 96.7%



```
eksperimen5 = tpot.TPOTClassifier(generations=10, population_size=100, verbosity=2)
eksperimen5.fit(X_train, y_train)
```

```
Imputing missing values in feature set

Generation 1 - Current best internal CV score: 0.9756756756756758

Generation 2 - Current best internal CV score: 0.9756756756756758

Generation 3 - Current best internal CV score: 0.9756756756756758

Generation 4 - Current best internal CV score: 0.981081081081081

Generation 5 - Current best internal CV score: 0.981081081081081

Generation 6 - Current best internal CV score: 0.981081081081081

Generation 7 - Current best internal CV score: 0.981081081081081

Generation 8 - Current best internal CV score: 0.981081081081081

Generation 9 - Current best internal CV score: 0.981081081081081

Generation 10 - Current best internal CV score: 0.981081081081081

Best pipeline: MLPClassifier(PCA(RobustScaler(input_matrix), iterated_power=6, svd_sol
```

```
TPOTClassifier
TPOTClassifier(generations=10, verbosity=2)
```

```
[ ] #Menlihat akurasi
print("Accuracy is {}%".format(eksperimen5.score(X_test, y_test)*100))
```

```
Imputing missing values in feature set
Accuracy is 95.6989247311828%
```

Eksperimen kelima dengan mengatur parameter generation=10 dan population size=100. Didapatkan hasil best CV score sebesar 0.981 serta akurasi pada data test sebesar 95.6%. Berdasarkan hasil eksperimen yang dilakukan, didapatkan parameter optimal dengan mengatur generation=5 dan population size=100. Didapatkan hasil CV Score sebesar 0.983 serta akurasi pada data test sebesar 96.7%. Dengan best pipeline yaitu MLPClassifier.

BAB V KESIMPULAN

Berdasarkan project dan eksperimen yang sudah dilakukan, dapat disimpulkan bahwa metode Automated Machine Learning merupakan salah satu metode pengolahan data yang dapat mengotomasi beberapa tahapan dalam melakukan pengolahan serta pemodelan data. Auto Machine Learning dapat secara otomatis menentukan model terbaik untuk mengolah data sesuai dengan datasets yang digunakannya.

Terdapat beberapa hal yang dapat mempengaruhi hasil dari pemodelan menggunakan metode Automated Machine Learning ini. Diantaranya adalah parameter yang digunakan pada Automated Machine Learning akan menentukan hasil serta model terbaik, selanjutnya adalah datasets yang digunakan juga berpengaruh terhadap hasil dari pemodelan menggunakan Automated Machine Learning ini. Diperlukannya beberapa eksperimen atau percobaan terhadap parameter yang digunakan untuk mendapatkan hasil yang maksimal dari pemodelan ini.

Berdasarkan eksperimen yang sudah dilakukan, didapatkan hasil paling optimal dengan mengatur generation=5 dan population size=100. didapatkan CV score sebesar 0.983 serta akurasi pada data test sebesar 96.7%. Serta best pipeline dengan menggunakan MLP Classifier.

Link Collab:

https://colab.research.google.com/drive/12NjYojWEcMOQ6vUpSnwy17aUEbL2_bCk?usp=sharing