Feature_Selection Having irrelevant features in your data can decrease the accuracy of the models and makes your models learn based on irrelevant

Defination Feature Selection:

- Process of selecting the best features which contribute maximum for the model in order to get best result in term of accuracy or it should take less time for traning.

In [4]:

Out[6]:

- **Benefits of Performing Feature-Selection:**

- model in order to predict the target variable.
- Feature selection methods are intended to reduce the number of input variables to those that are believed to be most useful to a

- 1. Reduce Overfitting Improve Accuracy Reduce Traning Time

 - import numpy as np import pandas as pd import matplotlib.pyplot as plt import seaborn as sns import warnings warnings.filterwarnings("ignore")

 - pwd

 - 'E:\\DataScience\\MachineLearning\\Breast-cancer-detection-using-ML'
 - path='E:\\DataScience\\MachineLearning\\Breast-cancer-detection-using-ML'

 - import os

 - os.listdir(path)
- ['.ipynb checkpoints', 'Breast_Cancer_Detection_Using ML.ipynb', 'Breast_Cancer_Detection_Using_ML.pdf', 'Breast_Cancer_Detection Using ML.py', 'data.csv',
- df =pd.read csv(path+"\\data.csv")
- 'Feature Selection.ipynb'] #reading data
- df

0

568

92751

569 rows × 33 columns

dtype='object')

df.columns

df.shape

diagnosis radius mean

area mean

radius_se

texture se perimeter se area se

smoothness_se compactness se concavity se

symmetry_se

radius worst texture_worst perimeter worst area worst

Unnamed: 32

0

1

2

3

4

564

565

566

567

568

In [14]:

dtype: int64

concave points_se

smoothness worst compactness worst concavity_worst concave points worst symmetry worst

fractal_dimension_se

fractal_dimension_worst

Μ

M

M

Μ

Μ

Μ

В

569 rows × 31 columns

X = df.iloc[:,1:]#target values y = df.iloc[:,0]

using chi2 function

#X chi2 selected

23

3 13

22

2

20

0

12

21

1 26

6

22 7

20

2

23 0 3

10

12 13

fs chi2 =SelectKBest(chi2 , k=15) X_chi2 = fs_chi2.fit_transform(X,y)

fs_f =SelectKBest(f_classif , k=15) X_f_classif = fs_f.fit_transform(X,y)

dfcolumns =pd.DataFrame(X.columns)

#naming the dataframe columns

dfscores = pd.DataFrame(fs chi2.scores)

featureScores.columns = ['Specs', 'Score'] print(featureScores.nlargest(15, 'Score'))

Specs

area se

perimeter worst

perimeter_mean

radius_worst

perimeter se

texture_worst

texture_mean

radius se

dfscores = pd.DataFrame(fs f.scores) dfcolumns =pd.DataFrame(X.columns)

featureScores.columns = ['Specs', 'Score'] print(featureScores.nlargest(15, 'Score'))

Specs

area_worst 661.600206 radius_mean 646.981021 area_mean 573.060747

#naming the dataframe columns

27 concave points_worst 964.385393 perimeter worst 897.944219

6 concavity_mean 533.793126 26 concavity_worst 436.691939 5 compactness_mean 313.233079 25 compactness_worst 304.341063

Feature Importance

import matplotlib.pyplot as plt model = ExtraTreesClassifier()

model.fit(X,y)

plt.show()

radius_mean concavity_worst perimeter_mean concave points_mean concavity_mean perimeter_worst area_mean area_worst radius_worst

concave points_worst

using the seaborn library.

top_corr_features = corr.index plt.figure(figsize=(20,20))

corr = df.corr()

#plot heat map

fractal dimension mean

radius_se 268.840327

perimeter_se 253.897392 area_se 243.651586

Feature importance gives you a score for each feature of your data,

from sklearn.ensemble import ExtraTreesClassifier

Feature importance is an inbuilt class that comes with Tree Based Classifiers,

 $[0.04517047 \ 0.01606722 \ 0.05111449 \ 0.08301556 \ 0.00874529 \ 0.0233687]$ 0.06424779 0.05671327 0.00715549 0.00531724 0.01738561 0.00494817 $0.01884134 \ 0.03305324 \ 0.00577992 \ 0.00600593 \ 0.00966862 \ 0.0119452$ $0.00600117 \ 0.00729472 \ 0.0995078 \ \ 0.02793595 \ 0.07422834 \ 0.08355751$ $0.01967976 \ 0.02634849 \ 0.04936704 \ 0.10803962 \ 0.01739662 \ 0.01209941]$

#plot graph of feature importances for better visualization

feat_importances.nlargest(10).plot(kind='barh')

Correlation Matrix with Heatmap

value of feature decreases the value of the target variable)

#get correlations of each features in dataset

Correlation states how the features are related to each other or the target variable.

plt=sns.heatmap(df[top_corr_features].corr(),annot=True,cmap="RdYlGn")

feat_importances = pd.Series(model.feature_importances_, index=X.columns)

The higher the score more important or relevant is the feature towards your output variable.

print (model.feature importances) #use inbuilt class feature importances of tree based classifiers

Correlation can be positive (increase in one value of feature increases the value of the target variable) or negative (increase in one

Heatmap makes it easy to identify which features are most related to the target variable, we will plot heatmap of correlated features

1 0.33 0.32 0.023 0.24 0.3 0.29 0.071 0.076 0.28 0.39 0.28 0.26 0.0066 0.19 0.14 0.16 0.00910.054 0.35 0.91 0.36 0.34 0.078 0.28 0.3 0.3 0.1 0.12

1 0.33 1 0.99 0.21 0.56 0.72 0.85 <mark>0.18 0.26</mark> 0.69 <mark>0.087</mark> 0.69 0.74 <mark>0.2 0.25 0.23 0.41 0.0820.005</mark>50.97 0.3 0.97 0.94 0.15 0.46 0.56 0.77 0.19 0.051

0.99 0.32 0.99 1 0.18 0.5 0.69 0.82 0.15 0.28 0.73 0.066 0.73 0.066 0.73 0.8 0.17 0.21 0.21 0.37 0.072 0.02 0.96 0.29 0.96 0.96 0.12 0.39 0.51 0.72 0.14 0.00

-0.31 -0.076 -0.26 -0.28 0.58 0.57 0.34 0.17 0.48 1 0 0001 0.16 0.04 -0.09 0.4 0.56 0.45 0.34 0.35 0.69 -0.25 -0.051 -0.21 -0.23 0.5 0.46 0.35 0.18 0.33 0.73

0.097 0.39 0.087-0.066 0.068 0.046 0.076 0.021 0.13 0.16 0.21 1 0.22 0.11 0.4 0.23 0.19 0.23 0.41 0.28 0.11 0.41 -0.1 0.083 0.074 0.092 0.069 0.12 -0.13 0.04

0.74 0.26 0.74 0.8 0.25 0.46 0.62 0.69 0.22 0.09 0.95 0.11 0.94 1 0.075 0.28 0.27 0.42 0.13 0.13 0.76 0.2 0.76 0.81 0.13 0.28 0.39 0.54 0.074 0.01

-0.220,0066 -0.2 -0.17 0.33 0.14 0.099 0.028 0.19 0.4 0.16 0.4 0.15 0.075 1 0.34 0.27 0.33 0.41 0.43 -0.23 -0.075 -0.22 -0.18 0.31 -0.056 0.058 -0.1 -0.11 0.1

-0.1 0.00910.082-0.072 0.2 0.23 0.18 0.095 0.45 0.35 0.24 0.41 0.27 0.13 0.41 0.39 0.31 0.31 1 0.37 -0.13 0.077 -0.1 -0.11 -0.013 0.06 0.037 -0.03 0.39 0.078 0.0430.0540.00550.02 0.28 0.51 0.45 0.26 0.33 0.69 0.23 0.28 0.24 0.13 0.43 0.8 0.73 0.61 0.37 1 0.0370.00320.0010.023 0.17 0.39 0.38 0.22 0.11 0.59

-0.4

-0.2

0.0

-0.2

smoothness_mean - 0.17 0.023 0.21 0.18 1 0.66 0.52 0.55 0.56 0.58 0.3 0.068 0.3 0.25 0.33 0.32 0.25 0.38 0.2 0.25 0.38 0.2 0.28 0.21 0.036 0.24 0.21 0.81 0.47 0.43 0.5 0.39 0.5

compactness_mean - 0.51 0.24 0.56 0.5 0.66 1 0.88 0.83 0.6 0.57 0.5 0.046 0.55 0.46 0.14 0.74 0.57 0.64 0.23 0.51 0.54 0.25 0.59 0.51 0.57 0.87 0.82 0.82 0.82 0.51 0.69

concave points_mean - 0.82 0.29 0.85 0.82 0.55 0.83 0.92 1 0.46 0.17 0.7 0.021 0.71 0.69 0.028 0.49 0.44 0.62 0.095 0.26 0.83 0.29 0.86 0.81 0.45 0.67 0.75 0.91 0.38 0.37

concavity mean - 0.68 0.3 0.72 0.69 0.52 0.88 1 0.92 0.5 0.34 0.63 0.076 0.66 0.62 0.099 0.67 0.69 0.68 0.18 0.45 0.69 0.3 0.73 0.68 0.45 0.75 0.88 0.86 0.41 0.51

symmetry_mean - 0.15 0.071 0.18 0.15 0.56 0.6 0.5 0.46 1 0.48 0.3 0.13 0.22 0.19 0.42 0.34 0.39 0.45 0.33 0.19 0.091 0.22 0.18 0.43 0.47 0.43 0.43 0.7 0.44

radius se - 0.68 0.28 0.69 0.73 0.3 0.5 0.63 0.7 0.3 0.00011 1 0.21 0.97 0.95 0.16 0.36 0.33 0.51 0.24 0.23 0.72 0.19 0.72 0.75 0.14 0.29 0.38 0.53 0.095 0.05

perimeter_se - 0.67 0.28 0.69 0.73 0.3 0.5 0.66 0.71 0.31 0.04 0.97 0.22 1 0.94 0.15 0.42 0.36 0.56 0.27 0.24 0.7 0.2 0.72 0.73 0.13 0.34 0.42 0.55 0.11 0.085

compactness_se - 0.21 0.19 0.25 0.21 0.32 0.74 0.67 0.49 0.42 0.56 0.36 0.23 0.42 0.28 0.34 1 0.8 0.74 0.8 0.74 0.8 0.74 0.8 0.74 0.8 0.2 0.14 0.26 0.2 0.23 0.68 0.64 0.48 0.28 0.59

concave points_se - 0.38 0.16 0.41 0.37 0.38 0.64 0.68 0.62 0.39 0.34 0.51 0.23 0.56 0.42 0.33 0.74 0.77 1 0.31 0.61 0.36 0.087 0.39 0.34 0.22 0.45 0.55 0.6 0.14 0.31

concavity se - 0.19 0.14 0.23 0.21 0.25 0.57 0.69 0.44 0.34 0.45 0.33 0.19 0.36 0.27 0.27 0.8 1 0.77 0.31 0.73 0.19 0.1 0.23 0.19 0.17 0.48 0.66 0.44 0.2 0.44

texture_worst - 0.3 0.91 0.3 0.29 0.036 0.25 0.3 0.29 0.091 0.051 0.19 0.41 0.2 0.2 0.075 0.14 0.1 0.0870.0770.003 0.36 1 0.37 0.35 0.23 0.26 0.37 0.36 0.23 0.22

compactness_worst - 0.41 0.28 0.46 0.39 0.47 0.87 0.75 0.67 0.47 0.46 0.29 0.092 0.34 0.28 0.056 0.68 0.48 0.45 0.06 0.39 0.48 0.36 0.53 0.44 0.57 1 0.89 0.8 0.61 0.81

concave points_worst - 0.74 0.3 0.77 0.72 0.5 0.82 0.86 0.91 0.43 0.18 0.53 0.12 0.55 0.54 0.1 0.48 0.44 0.6 0.03 0.22 0.79 0.36 0.82 0.75 0.55 0.8 0.86 1

texture_mea

texture_mea

perimeter_mea

area_me

smoothness_ma

compactness_ma

concavity_m

concave points_n

symmetry_n

fractal_dimension_r

symmetry worst - 0.16 0.11 0.19 0.14 0.39 0.51 0.41 0.38 0.7 0.33 0.095 0.13 0.11 0.074 0.11 0.28 0.2 0.14 0.39 0.11 0.24 0.23 0.27 0.21 0.49 0.61 0.53 0.5

.0071 0.12 0.0510 0037 0.5 0.69 0.51 0.37 0.44 0.77 0.05 0.0460.085 0.018 0.1 0.59 0.44 0.31 0.078 0.59 0.093 0.22 0.14 0.08 0.62 0.81 0.69 0.51 0.54

radius_se texture_se area_se area_se area_se compactness_se
concavity_se
concave points_se
symmetry_se
fractal_dimension_se
radius_wors
texture_wors
perimeter_wor
area_wor
compactness_wo
compactness_wo
compactness_wo
concavity_wc
concave points_w

area_worst - 0.94 0.34 0.94 0.96 0.21 0.51 0.68 0.81 0.18 0.23 0.75 0.083 0.73 0.81 0.81 0.18 0.22 0.19 0.34 0.11 0.023 0.98 0.35 0.98 1 0.21 0.44 0.54 0.75 0.21 0.08

• We will be using Extra Tree Classifier for extracting the top 10 features for the dataset.

concave points mean 861.676020

radius_worst 860.781707 perimeter_mean 697.235272

concavity_worst

concavity_mean

25 compactness_worst 27 concave points_worst

#X f classif Selected

radius mean

featureScores = pd.concat([dfcolumns,dfscores],axis=1)

area_worst 112598.431564

area_mean 53991.655924

featureScores = pd.concat([dfcolumns,dfscores],axis=1)

using f classif function

texture mean perimeter_mean

smoothness mean

compactness mean concavity_mean concave points mean symmetry mean

fractal dimension_mean

df.isnull().sum()

Out[10]: (569, 33)

Out[11]: id

- id diagnosis radius_mean texture_mean perimeter_mean area_mean smoothness_mean compactness_mean concavity_mean 842302 Μ 17.99
- 842517 Μ 84300903 Μ
- 20.57 19.69 84348301 11.42 **4** 84358402 M 20.29
- 926424 564 Μ
- 565 926682 926954 566 Μ 567
 - 927241
 - 21.56 В
 - 20.13 28.25 16.60 20.60 7.76

0

0 0

0

0

0

0

0

0

0

0

10.38

17.77

21.25

20.38

14.34

22.39

28.25

28.08

29.33

24.54

569

df=df.drop(['Unnamed: 32','id'],axis=1)

17.99

20.57

19.69

11.42

20.29

21.56

20.13

16.60

20.60

7.76

Univariate feature selection

#importing feauture selection from sklearn

#here we are using data shape -- 569 , 32

from sklearn.feature selection import SelectKBest from sklearn.feature_selection import chi2 ,f_classif

#traning data -- all features except the target values

k= 15 : selecting top 15 values which are highly co-related to taget

k= 15 : selecting top 15 values which are highly co-related to taget

Score

8758.504705

3665.035416

2011.102864

491.689157

266.104917

250.571896

174.449400

93.897508

39.516915 34.675247

19.712354

19.314922 13.485419

Score

stastistical fuction --- chi

- 28.08
 - 29.33 24.54

10.38

17.77

21.25

20.38

14.34

22.39

- Out[9]: 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',

122.80

132.90

130.00

77.58

135.10

142.00

131.20

108.30

140.10

47.92

1001.0

1326.0

1203.0

386.1

1297.0

1479.0

1261.0

858.1

1265.0

181.0

0.11840

0.08474

0.10960

0.14250

0.10030

0.11100

0.09780

0.08455

0.11780

0.05263

0.27760

0.07864

0.15990

0.28390

0.13280

0.11590

0.10340

0.10230

0.27700

0.04362

0.30010

0.08690

0.19740

0.24140

0.19800

0.24390

0.14400

0.09251

0.35140

0.00000

concave

0.14710

0.07017

0.12790

0.10520

0.10430

0.13890

0.09791

0.05302

0.15200 0.00000

points_mean

0.30010

0.08690

0.19740

0.24140

0.19800

0.24390

0.14400

0.09251

0.35140

0.00000

- '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', 'Unnamed: 32'],

diagnosis radius_mean texture_mean perimeter_mean area_mean smoothness_mean compactness_mean concavity_mean

1001.0

1326.0

1203.0

386.1

1297.0

1479.0

1261.0

858.1

1265.0

181.0

0.11840

0.08474

0.10960

0.14250

0.10030

0.11100

0.09780

0.08455

0.11780

0.05263

0.27760

0.07864

0.15990

0.28390

0.13280

0.11590

0.10340

0.10230

0.27700

0.04362

122.80

132.90

130.00

77.58

135.10

142.00

131.20

108.30

140.10

47.92

For regression: f_regression, mutual_info_regression For classification: chi2, f_classif, mutual_info_classif