



★ **Last chance:** Become a member and get 25% off the first year (Ends 1/31)



Feature Selection for Machine Learning Models using Iris dataset



Kousik Roy · Follow

9 min read · Jul 22, 2022



Listen



Share



More

Which features I need to use to create machine learning model?

Author : Kousik Roy Date : 10-Jul-2022 Version : 1.0

Load data

Iris flowers are distinguished by their Petal length & width and Sepal length & width.

I shall use this data to

1. find most influential feature (out of four features: Petal length, Petal width, Sepal Length and Sepal width)
2. create a logistic regression model
3. study the impact of accuracy of the machine learning model depending on the selection of feature or features.

This is an interesting data set to understand the concept of **feature selection in machine learning model**. I am excited to go through the steps below.

Package to load Iris data.

```
1. import pydataset
```

Store the data to df variable. This data set is stored as pandas dataframe.

```
1. df=pydataset.data('iris')
```

View data set

Inspect the head of the dataset.

```
1. df.head()
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa

Name of the columns for this data set have “.” in them. I want to clean up the column name and rename them with a smaller names.

```
1. df=df.rename(columns=
({'Sepal.Length':'SepLength','Sepal.Width':'SepWidth','Petal.Length':'PetLength','Petal.Width':'PetWidth'}))
```

Let me view the change of column names.

```
1. df.head()
```

	SepLength	SepWidth	PetLength	PetWidth	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa

Visualize features

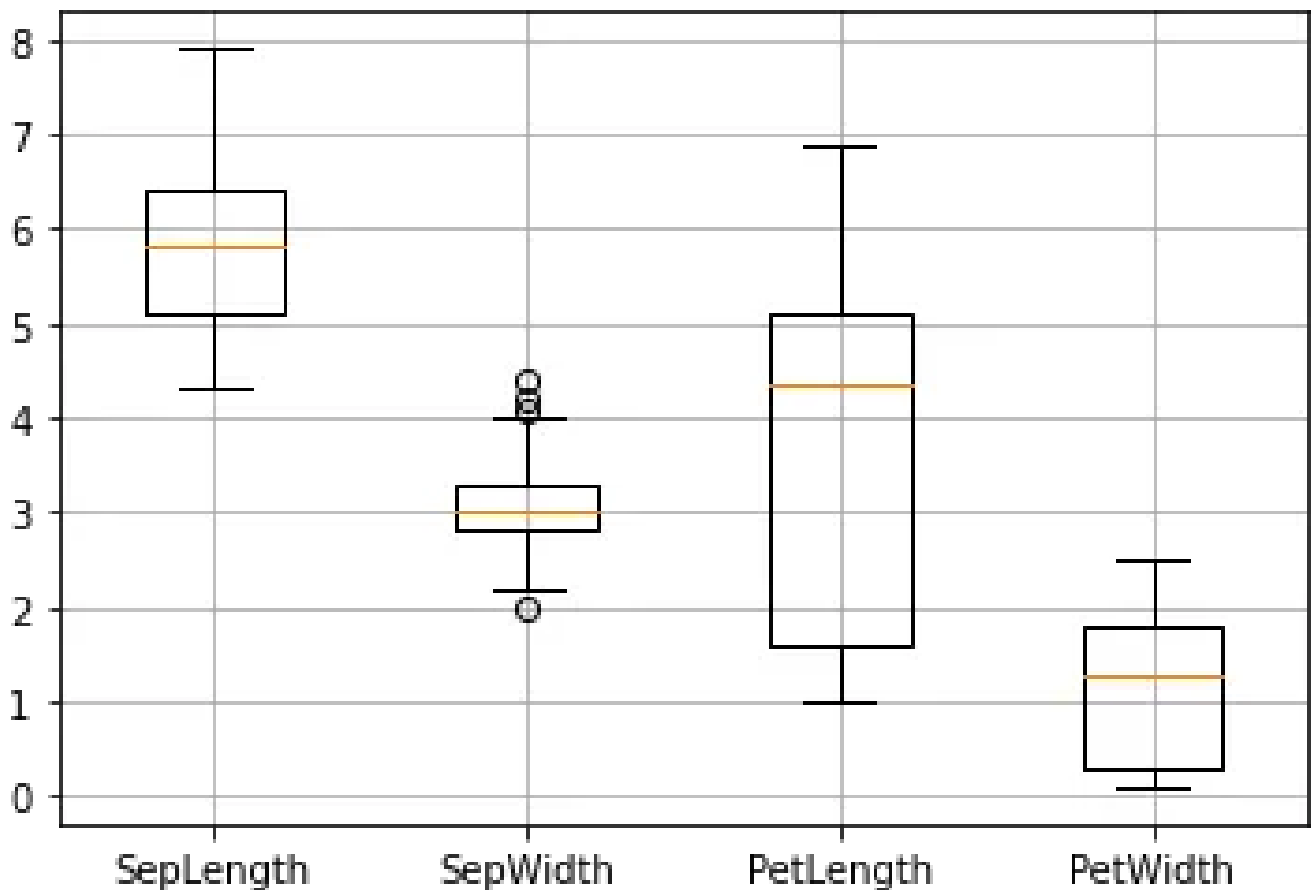
I shall visualize the values of each features to see the range and distribution of values. This visualization will help me in the next steps to select a suitable feature for my machine learning model.

Keeping the name of the columns in a variable. I shall use it for marking x-axis in visualization.

```
1. x_names=list(df.columns[:4])  
2. from matplotlib import pyplot as plt
```

- Select only first four columns as the last one is for the Species.
- Get values in the form of an array.
- Draw a box plot values of all features.
- Mark the tick points at x-axis with the name of the features.
- Draw grids on the plot.
- Draw the entire box plot with data and xticks with the use of plt.show().

```
1. plt.boxplot(df.iloc[:,0:4].values);  
2. plt.xticks([1,2,3,4],x_names);  
3. plt.grid();  
4. plt.show();
```



My visual observation are

1. Sepal length is in the range of 5 to 6.5
2. Sepal width is around 3
3. Petal length is in the range of 1.5 to 5.2
4. Petal width is in the range of 0.5 to 1.8

Any values other than this range are considered as outliers. I can observe few outliers in ir Sepal Width. Let me observe those data points in next step.

Outliers

Values of first, second(median) and 3rd quantile for the sepal width are.

```
1. SepWidth_quantile=df.SepWidth.quantile([0.25,0.5,0.75])
2. SepWidth_quantile
```

Output:

```
0.25    2.8
0.50    3.0
0.75    3.3
Name: SepWidth, dtype: float64
```

Inter quantile range

```
1. SepWidth_iqr = SepWidth_quantile[0.75] - SepWidth_quantile[0.25]
2. SepWidth_iqr
```

Output:

```
0.5
```

Find upper and lower bound

```
1. SepWidth_upper = SepWidth_quantile[0.75] + 1.5 * SepWidth_iqr
2. SepWidth_lower = SepWidth_quantile[0.25] - 1.5 * SepWidth_iqr
3. print(SepWidth_lower, SepWidth_upper)
```

Output:

```
2.05 4.05
```

Four records are outliers

```
1. df[(df.SepWidth <SepWidth_lower) | (df.SepWidth >SepWidth_upper)]
```

	SepLength	SepWidth	PetLength	PetWidth	Species
16	5.7	4.4	1.5	0.4	setosa
33	5.2	4.1	1.5	0.1	setosa
34	5.5	4.2	1.4	0.2	setosa
61	5.0	2.0	3.5	1.0	versicolor

Versicolor outliers

Before removing these outlier I shall check if for these two species 'setosa' and 'versicolor' these are really outlier considering each species separately.

Get quantiles of Sepal width for versicolor type of flower

```
1. versicolor_quantile =
df[df.Species=='versicolor'].SepWidth.quantile([0.25,.5,0.75])

2. versicolor_quantile
```

Output:

```
0.25    2.525
0.50    2.800
0.75    3.000
Name: SepWidth, dtype: float64
```

Get inter quantile range of sepal width for versicolor type of flower

```
1. versicolor_iqr=versicolor_quantile[0.75]-versicolor_quantile[0.25]

2. versicolor_iqr
```

Output:

```
0.47500000000000001
```

Get lower and upper bounday for sepal width for versicolor type of flower.

```
1. versicolor_lower = versicolor_quantile[0.25]-1.5*versicolor_iqr
2. versicolor_upper = versicolor_quantile[0.75]+1.5*versicolor_iqr
3. print(versicolor_lower, versicolor_upper)
```

Output:

```
1.8124999999999998 3.7125000000000004
```

I can observe that the outlier value of 2.0 found in previous step as a outlier is actually not an outlier when I consider only versicolor's values. Hence I shall keep this record.

Next we will perform this exercise for setosa flower type as below.

Setosa outliers

Get quantiles of Sepal width for setosa type of flower

```
1. setosa_quantile =  
df[df.Species=='setosa'].SepWidth.quantile([0.25,.5,0.75])  
  
2. setosa_quantile
```

Output:

```
0.25    3.200  
0.50    3.400  
0.75    3.675  
Name: SepWidth, dtype: float64
```

Get inter quantile range of sepal width for setosa type of flower

```
1. setosa_iqr=versicolor_quantile[0.75]-versicolor_quantile[0.25]  
2. setosa_iqr
```

Output:

```
0.47500000000000001
```

Get lower and upper bounday for sepal width for setosa type of flower.

```
1. setosa_lower = setosa_quantile[0.25]-1.5*setosa_iqr  
2. setosa_upper = setosa_quantile[0.75]+1.5*setosa_iqr  
3. print(setosa_lower, setosa_upper)
```

Output:

```
2.4875 4.3875
```

Though 4.4, 4.1, 4.2 values of sepal width were selected as outliers at earlier step, I can observe that only 4.4 is outside the setosa_lower and setosa_upper boundaries. Hence, I shall mark only 4.4 as outlier and remove it from data point. I must note here that removal of a record from analysis is one of the options to handle outliers. There are other options as well which is out of this analysis.

Outlier removal

I have 150 records before removal of the outlier

```
1. df.shape
```

Output:

```
(150, 5)
```

Select records which are below the upper boundary

```
1. df=df[df.SepWidth<=setosa_upper]
2. df.shape
```

Output:

```
(149, 5)
```

I have now 149 records for going forward with my analysis.

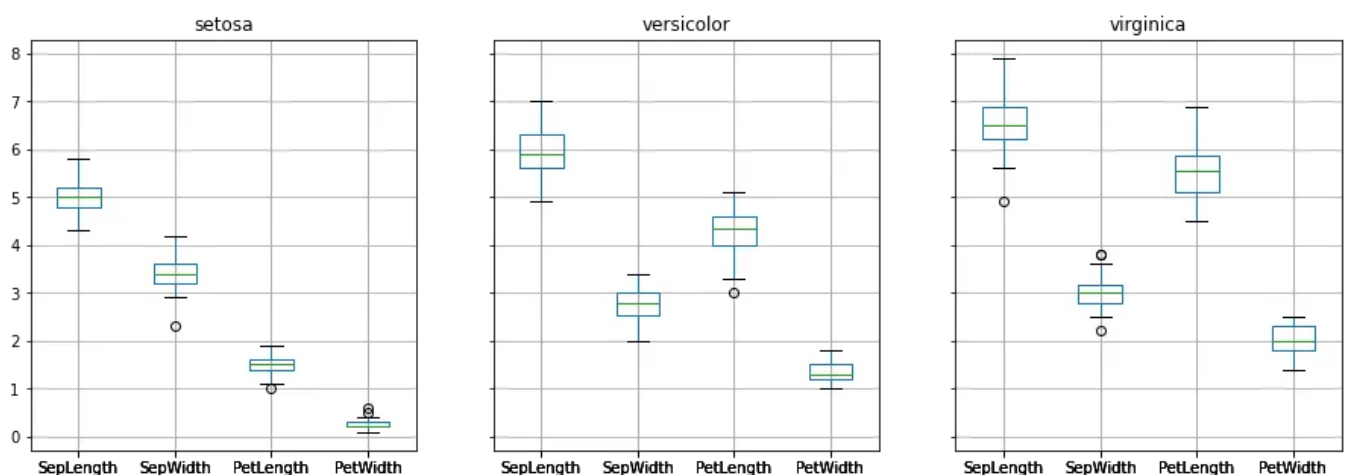
Species wise visualization

Enabling this warning-ignore option to stop displaying warnings. This will help me keeping outputs tidy.

```
1. import warnings
2. warnings.filterwarnings('ignore')
```

Define a figure with four subplots.

```
1. fig,ax_1 = plt.subplots(nrows=1,ncols=3,figsize=(15,5),sharey=True,sharex=True)
2. df.iloc[:,0:4].groupby(df.Species).boxplot(ax=ax_1);
3. plt.show();
```



From these box plots I can observe some outliers. Similar to previous outlier removal exercise I shall remove these outliers.

Setosa

Remove records for setosa type.

Sepal Width

```
1. qntl = df[df.Species == 'setosa'].SepWidth.quantile([0.25,0.5,0.75])
2. iqr = qntl[0.75]-qntl[0.25]
3. lower = qntl[0.25] - 1.5*iqr
4. upper = qntl[0.75] + 1.5*iqr
```

Remove the records of setosa which is outside the lower boundary. One record will be removed here.

```
1. df=df[~((df.Species == 'setosa')&(df.SepWidth<lower))]
```

Petal Length

```
1. qntl = df[df.Species == 'setosa'].PetLength.quantile([0.25,0.5,0.75])
2. iqr = qntl[0.75]-qntl[0.25]
3. lower = qntl[0.25] - 1.5*iqr
4. upper = qntl[0.75] + 1.5*iqr
```

Remove the records of setosa which is outside the lower boundary. One record will be removed here.

```
1. df=df[~((df.Species == 'setosa')&(df.PetLength<lower))]
```

Petal Width

```
1. qntl = df[df.Species == 'setosa'].PetWidth.quantile([0.25,0.5,0.75])
2. iqr = qntl[0.75]-qntl[0.25]
3. lower = qntl[0.25] - 1.5*iqr
4. upper = qntl[0.75] + 1.5*iqr
```

Remove the records of setosa which is outside the upper boundary. Two records will be removed here.

```
1. df=df[~((df.Species == 'setosa')&(df.PetWidth>upper))]
```

Versicolor

Remove records for versicolor type.

PetLength

```
1. qntl = df[df.Species == 'versicolor'].PetLength.quantile([0.25,0.5,0.75])
2. iqr  = qntl[0.75]-qntl[0.25]
3. lower = qntl[0.25] - 1.5*iqr
4. upper = qntl[0.75] + 1.5*iqr
```

Remove the records of versicolor which is outside the lower boundary. One record will be removed here.

```
1. df=df[~((df.Species == 'versicolor')&(df.PetLength<lower))]
```

Virginica

Remove records for virginica type.

Sepal Length

```
1. qntl = df[df.Species == 'virginica'].SepLength.quantile([0.25,0.5,0.75])
2. iqr  = qntl[0.75]-qntl[0.25]
3. lower = qntl[0.25] - 1.5*iqr
4. upper = qntl[0.75] + 1.5*iqr
```

Remove the records of virginica which is outside the lower boundary. One record will be removed here.

```
1. df=df[~((df.Species == 'virginica')&(df.SepLength<lower))]
```

Number of records in data set as of now.

```
1. df.shape
```

Output:
(143, 5)

Remove records for virginica type.

Sepal Width

```

1. qntl = df[df.Species == 'virginica'].SepWidth.quantile([0.25,0.5,0.75])
2. iqr = qntl[0.75]-qntl[0.25]
3. lower = qntl[0.25] - 1.5*iqr
4. upper = qntl[0.75] + 1.5*iqr
5. print(lower,upper)

```

Output:

```
2.1999999999999993 3.8000000000000007
```

Find min and max values of SepWidth for virginica species

```

1. print('min=',df[(df.Species == 'virginica')]['SepWidth'].min())
2. print('max=',df[(df.Species == 'virginica')]['SepWidth'].max())

```

Output:

```
min= 2.2
max= 3.8
```

My observation are

1. Though earlier box plot for virginical species has shown outlier points for Sepal Width, these are not actual outliers.
2. Hence we will not remove any data points further.

Final count of records for analysis is

```
1. df.shape
```

Output:

```
(143, 5)
```

Number of records per species of the flower are as below.

```
1. df.Species.value_counts()
```

Output:

```

versicolor    49
virginica     49
setosa        45
Name: Species, dtype: int64

```

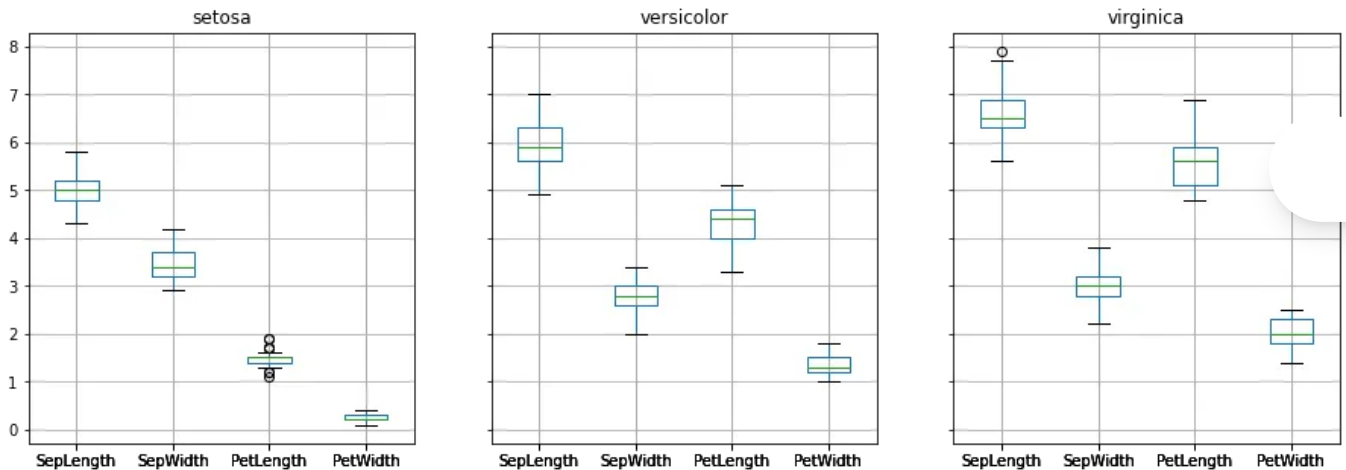
```

2. PetLength={'setosa':df[df.Species=='setosa']['PetLength'],
             'versicolor':df[df.Species=='versicolor']['PetLength'],
             'virginica':df[df.Species=='virginica']['PetLength']}

```

Re Draw the box plots after removing outliers

1. `fig,ax_1 = plt.subplots(nrows=1,ncols=3,figsize=(15,5),sharey=True,sharex=True)`
2. `df.iloc[:,0:4].groupby(df.Species).boxplot(ax=ax_1);`
3. `plt.show();`



Feature selection

From above box plots I can observe

1. Sepal length has overlapping values for versicolor and virginica
2. Sepal width has overlapping values for versicolor and virginica
3. Petal length has unique set of values for each species (there are no over lapping of values)
4. Petal Width also has unique set of values for each species (there may be a slight overlapping for versicolor and virginica).

From these observation I can choose 'Petal Length' and 'Petal Width' as the influential feature out of four features. For my analysis, I shall first choose the most influential feature 'Petal Length' and in the second iteration I shall include 'Petal Width' into my model.

One Feature (Petal Length)

Convert the categorical values of species to numerical index

1. `type_dict = {'setosa':0,'versicolor':1,'virginica':2}`
2. `df['Species']=[type_dict[i] for i in df['Species']]`

Get 80% of random row index from the entire data set. This indexes are for train dataset.

1. `df_train_index=df.sample(frac=0.8,random_state=123).index`

Rest of the indexes are for test dataset.

```
1. df_test_index=set(df.index).difference(set(df_train_index))
2. print('number of train records=',len(df_train_index),'| number of test
records=',len(df_test_index))
```

Output:

number of train records= 114 | number of test records= 29

Prepare train and test data sets

```
1. df_train=df.loc[df_train_index,:]
2. df_test=df.loc[df_test_index,:]
```

- Pick up only Petal length feature. Petal length feature is the 3rd column i.e. 2nd index of the array.
- Reshape of the numpy array is done to make sure that the X_train is a column vector and also y_train is a column vector
- Reshaping of data is required before feeding it to the machine learning package. However for different packages, the requirement of data format varies.

Train Data selection

```
1. X_train = df_train.values[:,2].reshape(-1,1)
2. y_train = df_train.values[:,4].reshape(-1,1)
```

Test Data selection

```
1. X_test = df_test.values[:,2].reshape(-1,1)
2. y_test = df_test.values[:,4].reshape(-1,1)
```

I shall use sklearn packages linear_model to create a logistic regression model

```
1. from sklearn.linear_model import LogisticRegression
```

Define object of logistic regression.

```
1. lr_Petal_Length=LogisticRegression(random_state=123)
```

Fit train data to the model. This fit() method will find optimal values of coefficients by multiple iterations.

```
1. lr_Petal_Length.fit(X_train,y_train)
2. LogisticRegression(random_state=123)
```

Train and Test accuracies

```
1. print('Train accuracy =',lr_Petal_Length.score(X_train,y_train), '| Test
Accuracy =',lr_Petal_Length.score(X_test,y_test))
```

Output:

Train accuracy = 0.9824561403508771 | Test Accuracy = 0.8620689655172413

This model is created using only one feature 'Petal Length'. With only one feature we get the test accuracy as 86%. This feature is the most influential to determine the type of species.

By choosing the most influential/important feature I have created a machine learning model which is simple but gives good accuracy. Next time some one approaches to me with data of Iris flower to identify its type, I shall tell him only to give me Petal Length and other features are not required.

This is the power of selecting correct feature.

coefficients of my model are as below.

```
1.
print('coefficient=',lr_Petal_Length.coef_,'\nintercept=',lr_Petal_Length.in
tercept_,'\nnnumber of iterations',lr_Petal_Length.n_iter_)
```

Output:

```
coefficient= [[-2.93788212]
[-0.29796821]
[ 3.23585033]]
intercept= [ 10.61315022  3.27076583 -13.88391605]
number of iterations [25]
```

Two Features (Petal Length and Petal Width)

Get Train data

```
1. X_train = df_train.values[:,2:4]
2. y_train = df_train.values[:,4].reshape(-1,1)
```

Get Test data

```
1. X_test = df_test.values[:,2:4]
2. y_test = df_test.values[:,4].reshape(-1,1)
```

Define class for logistic regression model

```
1. lr_petLength_width=LogisticRegression(random_state=123)
2. lr_petLength_width.fit(X_train,y_train)
```

Output:

```
LogisticRegression(random_state=123)
```

Model Accuracy

```
1. print('Train accuracy=',lr_petLength_width.score(X_train,y_train),'Test
accuracy=',lr_petLength_width.score(X_test,y_test))
```

Output:

```
Train accuracy= 0.9736842105263158 Test accuracy= 0.9655172413793104
```

Three Features (Petal Length & Width and Sepal Width)

Select Train data

```
1. X_train = df_train.values[:,1:4]
2. y_train = df_train.values[:,4].reshape(-1,1)
```

Select Test data

```
1. X_test = df_test.values[:,1:4]
2. y_test = df_test.values[:,4].reshape(-1,1)
```

Define logistic regression model

```
1. lr_petLength_petWidth_SepWidth=LogisticRegression(random_state=123)
```

```
2. lr_petLength_petWidth_SepWidth.fit(X_train,y_train)
```

Output:

```
LogisticRegression(random_state=123)
```

Model accuracy

```
1. print('Train  
accuracy=',lr_petLength_petWidth_SepWidth.score(X_train,y_train),'Test  
accuracy=',lr_petLength_petWidth_SepWidth.score(X_test,y_test))
```

Output:

```
Train accuracy= 0.9824561403508771 Test accuracy= 0.9655172413793104
```

All Features

Select Train data

```
1. X_train = df_train.values[:,0:4]  
2. y_train = df_train.values[:,4].reshape(-1,1)
```

Select Test data

```
1. X_test = df_test.values[:,0:4]  
2. y_test = df_test.values[:,4].reshape(-1,1)
```

Define logistic regression model

```
1. lr_all=LogisticRegression(random_state=123)  
2. lr_all.fit(X_train,y_train)
```

Output:

```
LogisticRegression(random_state=123)
```

Model Accuracy

```
1. print('Train accuracy=',lr_all.score(X_train,y_train),'Test  
accuracy=',lr_all.score(X_test,y_test))
```

Output:

```
Train accuracy= 0.9824561403508771 Test accuracy= 0.9655172413793104
```

Summary

The test accuracies of above models are

1. Only Petal Length 86.2%
2. Petal Length and Petal Width 96.5%
3. Petal Length & Width and Sepal Width 96.5%
4. With all features 96.5%

Machine learning model using only one feature 'Petal Length' achieves 86.2%. When the influential feature 'Petal Width' is included along with 'Petal Length', the test accuracy of the model increased to 96.5%. However adding more features does not improve the test accuracy.

I can conclude that only two features 'Petal Length' and 'Petal Width' are the important features. Hence to build a machine learning model I need to use only these two features.

End

Feature Selection

Machine Learning

Logistic Regression

Iris



Follow

Written by Kousik Roy 

63 Followers

Data Scientist and Machine learning practitioner, Published the book: Python for Data Analysts and Scientist:
<https://a.co/d/dOHbpnm>



More from Kousik Roy

calculated by USL (Upper specification limit), LSL (Lower specification limit), μ (mean of all values) and $\hat{\sigma}$ (sig). This $\hat{\sigma}$ is different from the normally calculated standard deviation (σ)

calculated as mean of range of all samples and a constant d_2 as $\frac{\bar{R}}{d_2}$.

red equations are:

$$\hat{\sigma} = \frac{\bar{R}}{d_2}$$
$$C_p = \frac{(USL - LSL)}{6 * \hat{\sigma}}$$
$$C_{pU} = \frac{(USL - \mu)}{3 * \hat{\sigma}}$$
$$C_{pL} = \frac{(\mu - LSL)}{3 * \hat{\sigma}}$$
$$C_{pK} = \min(C_{pU}, C_{pL})$$

 Kousik Roy 

CpK Calculation

3 min read · Jul 27, 2022



124




 Kousik Roy 

This is an excellent book for those who want to Jumpstart their career in Data Analytics and Data...

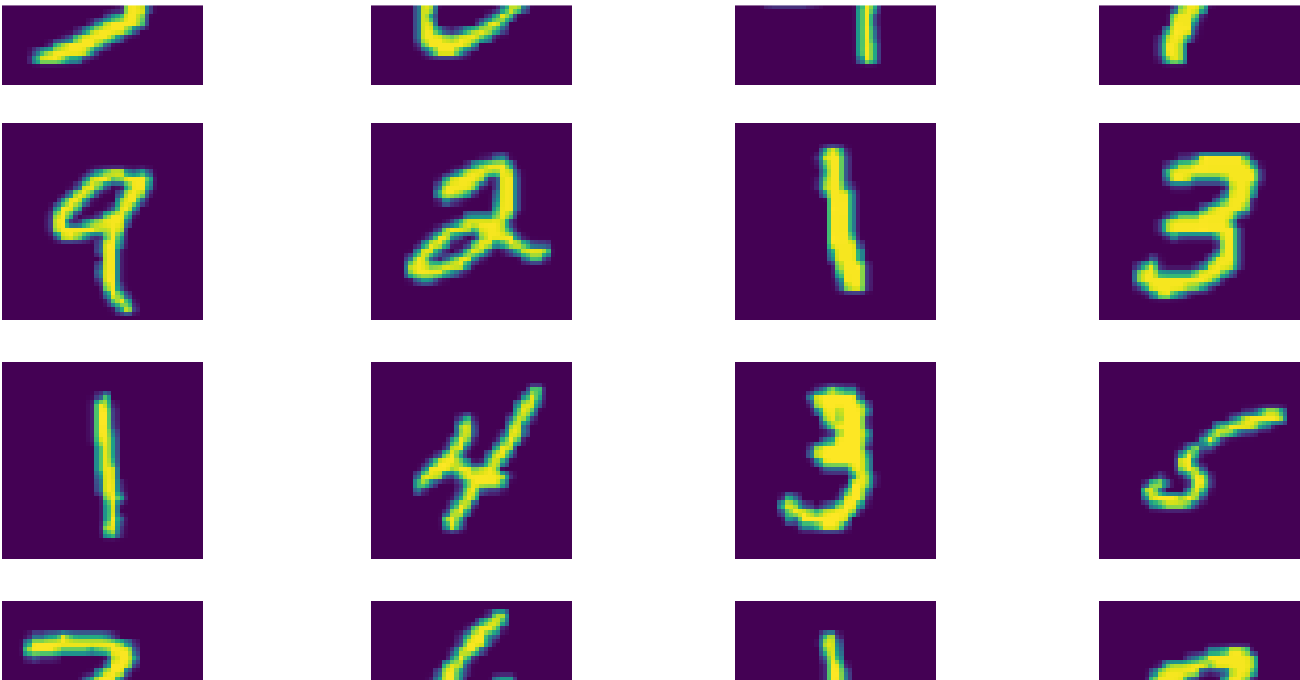
Motivation For Writing this book

High	Low	Open	Close	Volume
4.920013	321.089996	322.000000	323.570007	37000
9.980011	324.779999	326.100006	329.809998	44858
9.859985	325.529999	326.779999	325.899994	38069
6.359985	321.200012	323.119995	335.829987	56631
6.700012	330.299988	336.470001	330.750000	47032

 Kousik Roy 

Essential plots a Data Scientist must know

5 min read · Jul 20, 2022



Hand Written Image Identification by mean-image

7 min read · Jul 27, 2022

 199 

See all from Kousik Roy

...

Recommended from Medium





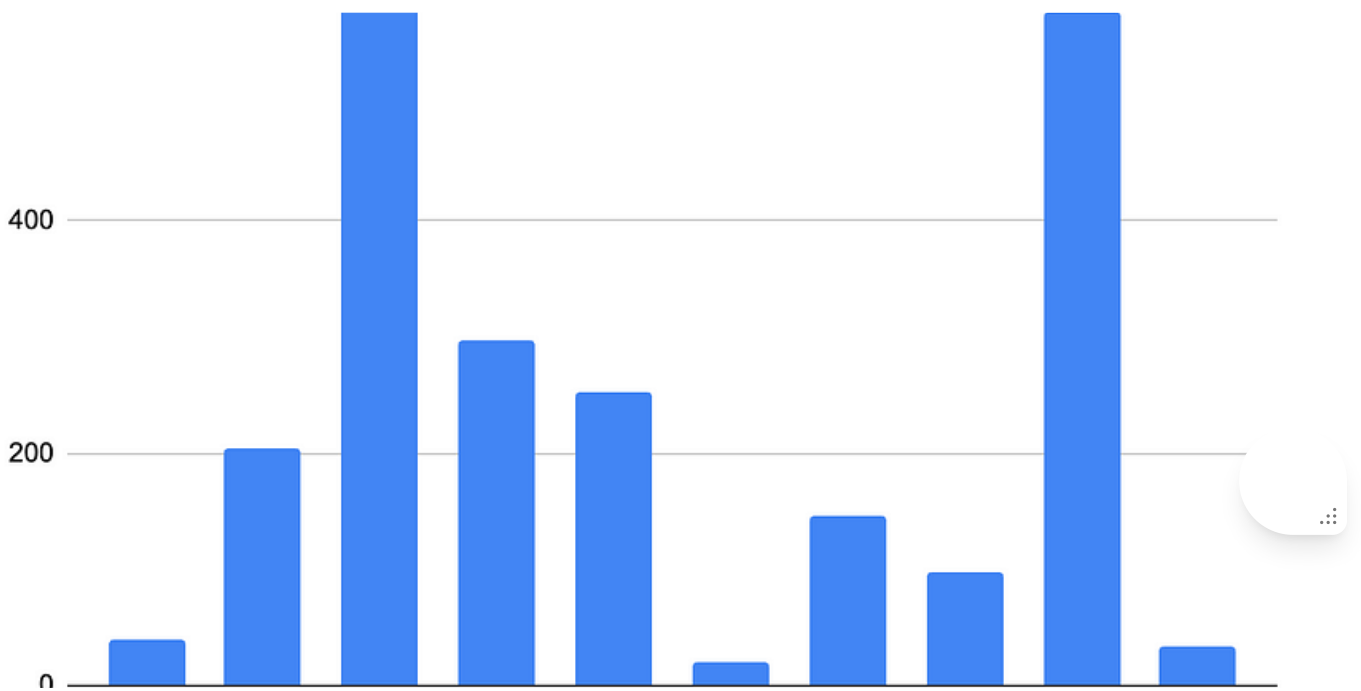
Deploying Machine Learning Models with Flask: A Step-by-Step Guide


Deploying machine learning models involves making your model accessible for others to use, typically through a web interface or an API...

★ · 2 min read · Nov 21, 2023

 8 



 Deepanjan Kundu in MLearning.ai

A Practical Guide for identifying important features using Python

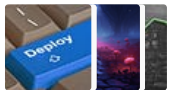
Feature Importance techniques such as SHAP, permutation importance and more

10 min read · Jul 31, 2023

 40 

Lists



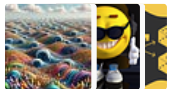
Predictive Modeling w/ Python

20 stories · 839 saves



Practical Guides to Machine Learning

10 stories · 979 saves



Natural Language Processing


1132 stories · 601 saves



The New Chatbots: ChatGPT, Bard, and Beyond

12 stories · 284 saves



 Tushar Aggarwal in Python in Plain English

LIME : Explaining Machine Learning Models with Confidence

Machine learning models have become increasingly complex and accurate over the years, but their opacity remains a significant challenge...

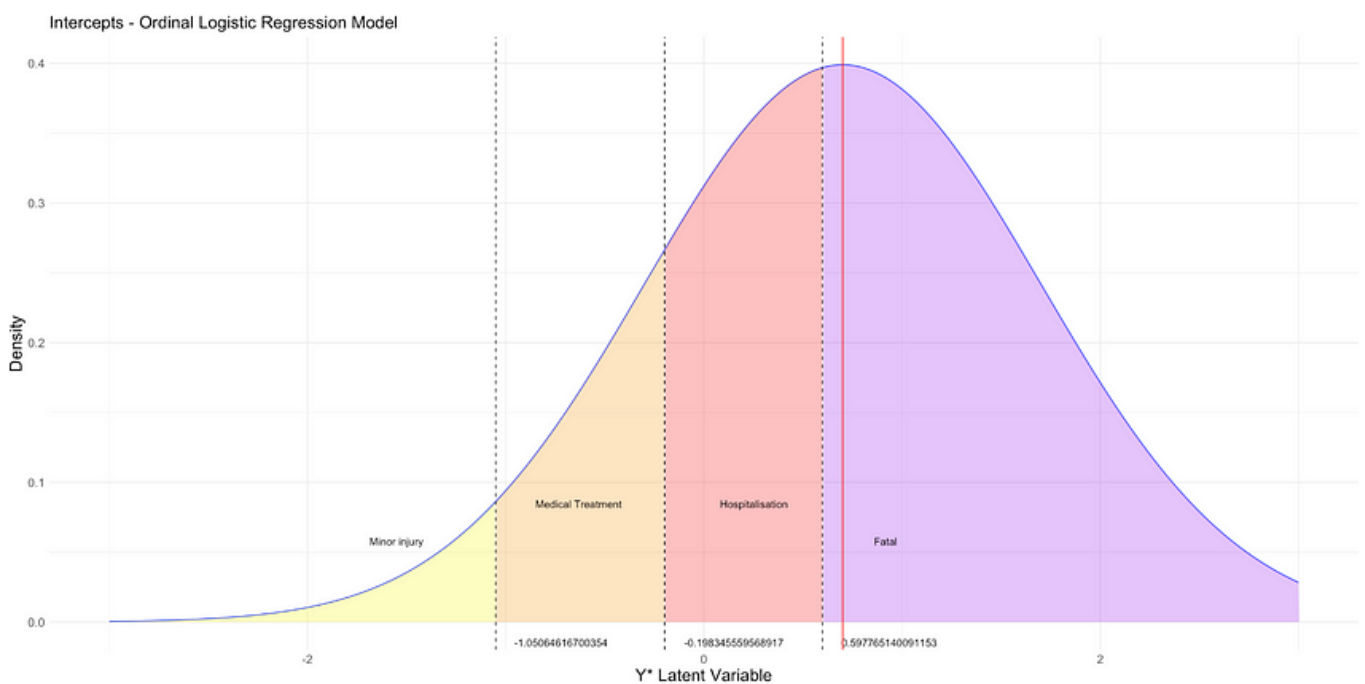
7 min read · Nov 3, 2023




114



1



 Carlos Poles

Ordinal Logistic Regression Analysis - Factors in Road Crash Severity in QLD


What atmospheric, lighting and road conditions increase the odds of getting involved in a more severe crash?

14 min read · Nov 13, 2023

 20 



 Aris Muhandisin

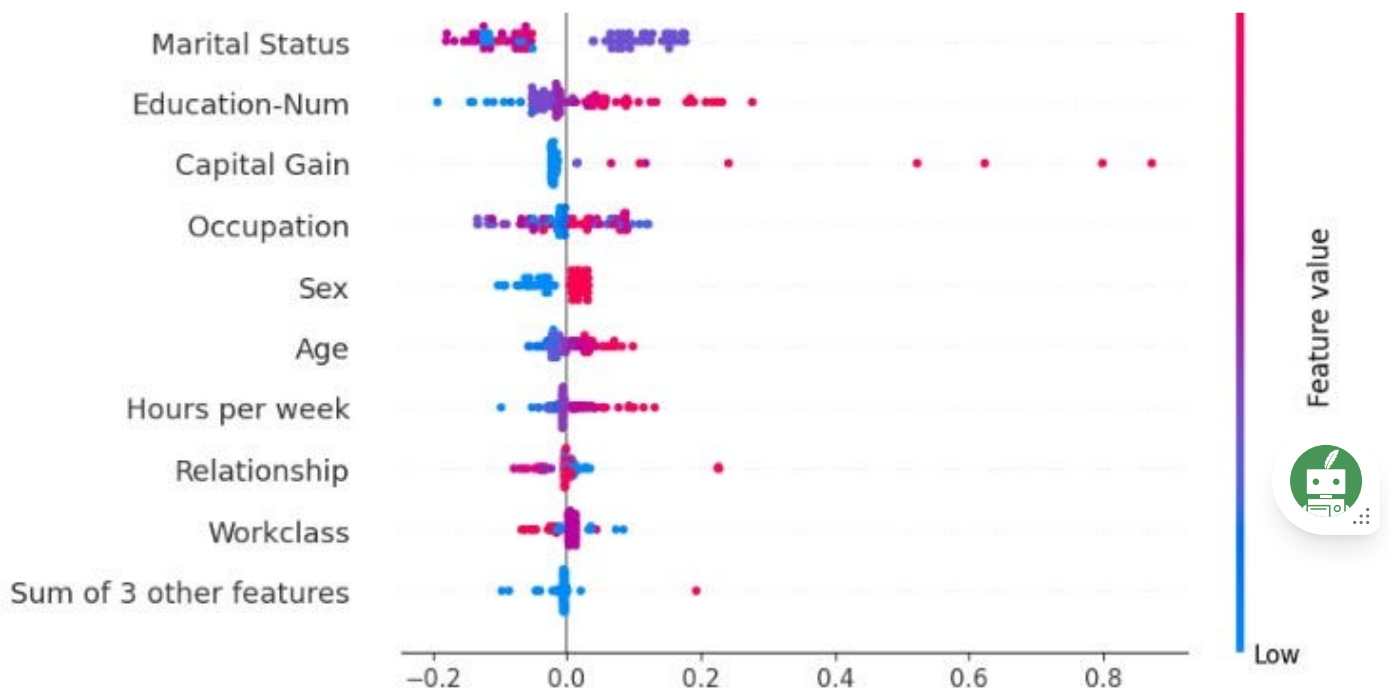
Understanding Wrapper Methods in Machine Learning: A Guide to Feature Selection


Machine learning algorithms can be incredibly powerful tools for making predictions and solving complex problems. However, their...

7 min read · Oct 12, 2023

 18 



 Alessandro Danesi in Data Reply IT | DataTech

Explainable AI: SHAP Values

Introduction

14 min read · Sep 25, 2023

 200 

See more recommendations