# Data Analysis

# Import libraries

In [86]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [87]:

```
import sklearn.datasets
```

In [88]:

```
iris = sklearn.datasets.load_iris()
```

In [89]:

```
print(iris)
```

```
- Iris-Setosa\n                - Iris-Versicolour\n                - Ir
is-Virginica\n                \n    :Summary Statistics:\n\n    =======
======= ==== ==== ======= ===== ====================\n
Min  Max   Mean    SD   Class Correlation\n    ============== ==== ====
======= ===== ====================\n    sepal length:   4.3  7.9   5.84
0.83    0.7826\n    sepal width:    2.0  4.4   3.05   0.43   -0.4194\n
petal length:   1.0  6.9   3.76   1.76    0.9490  (high!)\n    petal wi
dth:    0.1  2.5   1.20   0.76    0.9565  (high!)\n    ============== =
=== ==== ======= ===== ====================\n\n    :Missing Attribute V
alues: None\n    :Class Distribution: 33.3% for each of 3 classes.\n
:Creator: R.A. Fisher\n    :Donor: Michael Marshall (MARSHALL%PLU@io.ar
c.nasa.gov)\n    :Date: July, 1988\n\nThe famous Iris database, first u
sed by Sir R.A. Fisher. The dataset is taken\nfrom Fisher\'s paper. Not
e that it\'s the same as in R, but not as in the UCI\nMachine Learning
Repository, which has two wrong data points.\n\nThis is perhaps the bes
t known database to be found in the\npattern recognition literature.  F
isher\'s paper is a classic in the field and\nis referenced frequently
to this day.  (See Duda & Hart, for example.)  The\ndata set contains 3
classes of 50 instances each, where each class refers to a\ntype of iri
```

In [90]:

```
list(iris.target_names)
```

Out[90]:

```
['setosa', 'versicolor', 'virginica']
```

In [91]:

```
species = iris.target_names
```

In [92]:

```
list(iris.feature_names)
```

Out[92]:

```
['sepal length (cm)',
 'sepal width (cm)',
 'petal length (cm)',
 'petal width (cm)']
```

In [93]:

```
import pandas as pd
df = pd.DataFrame(iris.data, columns= iris.feature_names)
```

In [94]:

```
df['species'] = iris.target_names[iris.target]
```

In [95]:

```
df.head()
```

Out[95]:

| | sepal length (cm) | sepal width (cm) | petal length (cm) | petal width (cm) | species |
|---|---|---|---|---|---|
| **0** | 5.1 | 3.5 | 1.4 | 0.2 | setosa |
| **1** | 4.9 | 3.0 | 1.4 | 0.2 | setosa |
| **2** | 4.7 | 3.2 | 1.3 | 0.2 | setosa |
| **3** | 4.6 | 3.1 | 1.5 | 0.2 | setosa |
| **4** | 5.0 | 3.6 | 1.4 | 0.2 | setosa |

# data exploration

In [96]:

```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   sepal length (cm)  150 non-null    float64
 1   sepal width (cm)   150 non-null    float64
 2   petal length (cm)  150 non-null    float64
 3   petal width (cm)   150 non-null    float64
 4   species            150 non-null    object 
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

In [97]:

```
1 df.isnull().sum()
```

Out[97]:

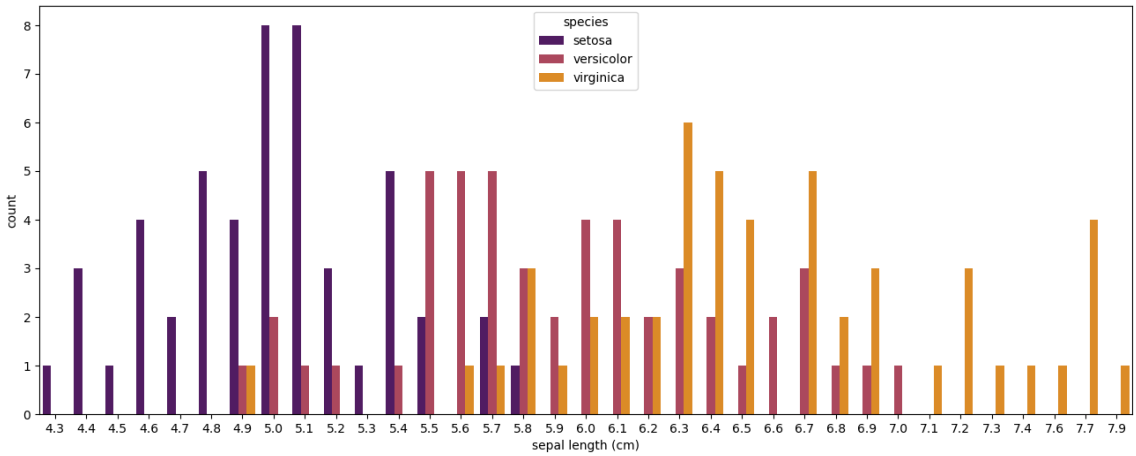
```
sepal length (cm)    0
sepal width (cm)     0
petal length (cm)    0
petal width (cm)     0
species              0
dtype: int64
```

# Visualizing diff columns

In [98]:

```
import seaborn as sns
plt.figure(figsize= (16,6))
sns.countplot(df['sepal length (cm)'],hue=df['species'],palette ='inferno')
plt.show()
```

```
C:\Users\Acer\anaconda37\lib\site-packages\seaborn\_decorators.py:36: Futu
reWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or misinterp
retation.
  warnings.warn(
```

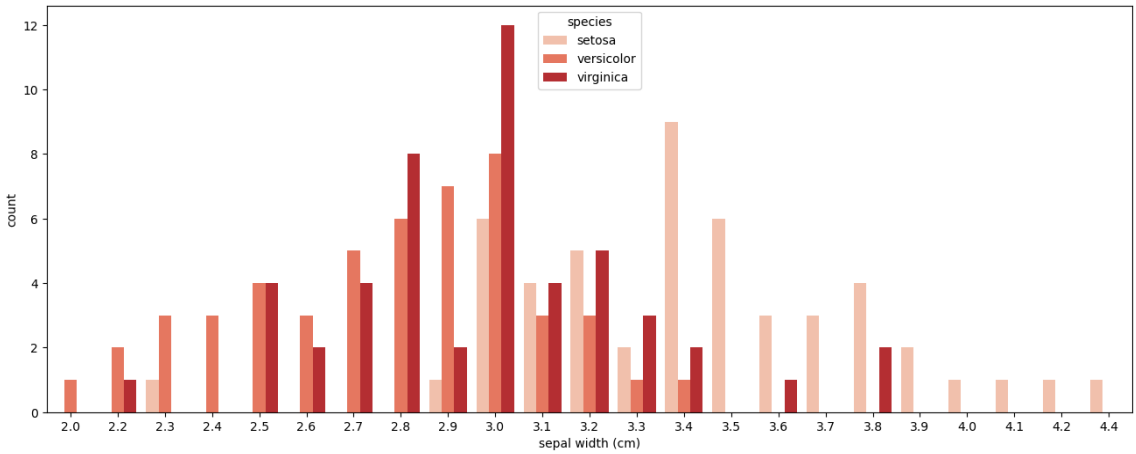



# Inference - Density of sentosa is more on lower sepal length and density of virginica is more on higher sepal length

In [99]:

```
import seaborn as sns
plt.figure(figsize= (16,6))
sns.countplot(df['sepal width (cm)'],hue=df['species'],palette ='Reds')
plt.show()
```

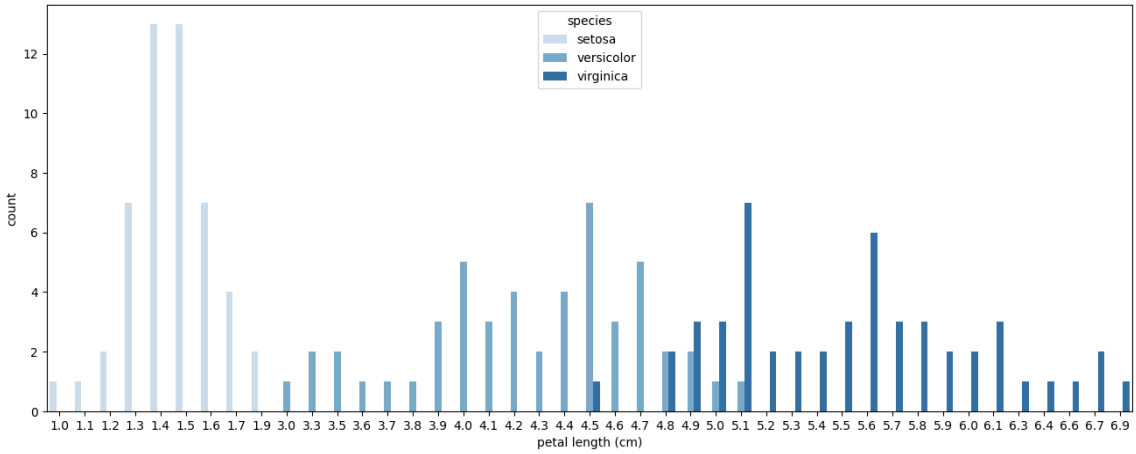
```
C:\Users\Acer\anaconda37\lib\site-packages\seaborn\_decorators.py:36: Futu
reWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or misinterp
retation.
  warnings.warn(
```




In [100]:

```
import seaborn as sns
plt.figure(figsize= (16,6))
sns.countplot(df['petal length (cm)'],hue=df['species'],palette ='Blues')
plt.show()
```

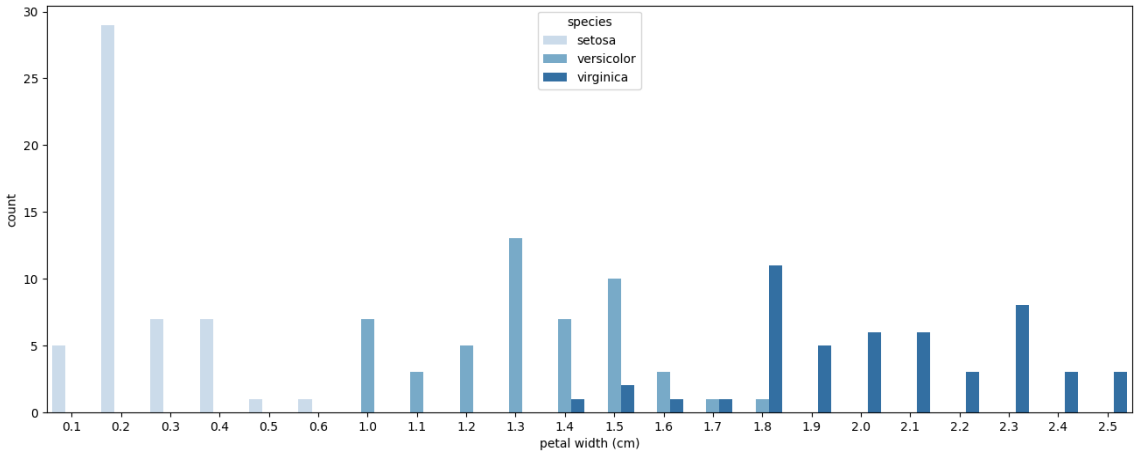
```
C:\Users\Acer\anaconda37\lib\site-packages\seaborn\_decorators.py:36: Futu
reWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or misinterp
retation.
  warnings.warn(
```



In [101]:

```
import seaborn as sns
plt.figure(figsize= (16,6))
sns.countplot(df['petal width (cm)'],hue=df['species'],palette ='Blues')
plt.show()
```

```
C:\Users\Acer\anaconda37\lib\site-packages\seaborn\_decorators.py:36: Futu
reWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or misinterp
retation.
  warnings.warn(
```




# Data Preprocessing

In [102]:

```
x = df.drop(['species'], axis = 1)
y = df['species']
```

In [103]:

```
x.shape
```

Out[103]:

```
(150, 4)
```

In [104]:

```
y.shape
```

Out[104]:

```
(150,)
```

# splitting dataset

In [105]:

```
from sklearn.model_selection import train_test_split
Xtrain, Xtest, ytrain, ytest = train_test_split(x, y , test_size=0.2, random_state=4
```

In [106]:

```
Xtrain.shape
```

Out[106]:

```
(120, 4)
```

In [107]:

```
 Xtest.shape
```

Out[107]:

```
(30, 4)
```

In [108]:

```
ytrain.shape
```

Out[108]:

```
(120,)
```

In [109]:

```
ytest.shape
```

Out[109]:

```
(30,)
```

In [110]:

```
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression()
```

from sklearn.datasets import make_classification from sklearn.linear_model import LogisticRegression from sklearn.model_selection import train_test_split from sklearn.pipeline import make_pipeline from sklearn.preprocessing import StandardScaler

pipe = make_pipeline(StandardScaler(), LogisticRegression()) pipe.fit(Xtrain, ytrain) # apply scaling on training data

pipe.score(Xtest, ytest) # apply scaling on testing data, without leaking training data.

In [111]:

```
lr.fit(Xtrain,ytrain)
lr.score(Xtrain,ytrain)
```

```
C:\Users\Acer\anaconda37\lib\site-packages\sklearn\linear_model\_logistic.
py:458: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown i
n:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://sc
ikit-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-reg
ression (https://scikit-learn.org/stable/modules/linear_model.html#logisti
c-regression)
  n_iter_i = _check_optimize_result(
```

Out[111]:

```
0.975
```

# Accuracy = 97%

# Model Evaluation

In [112]:

```
predictions = lr.predict(Xtest)
```

In [113]:

```
predictions
```

Out[113]:

```
array(['versicolor', 'setosa', 'virginica', 'versicolor', 'versicolor',
       'setosa', 'versicolor', 'virginica', 'versicolor', 'versicolor',
       'virginica', 'setosa', 'setosa', 'setosa', 'setosa', 'versicolor',
       'virginica', 'versicolor', 'versicolor', 'virginica', 'setosa',
       'virginica', 'setosa', 'virginica', 'virginica', 'virginica',
       'virginica', 'virginica', 'setosa', 'setosa'], dtype=object)
```

In [114]:

```
1 ytest
```

Out[114]:

```
73     versicolor
18         setosa
118     virginica
78     versicolor
76     versicolor
31         setosa
64     versicolor
141     virginica
68     versicolor
82     versicolor
110     virginica
12         setosa
36         setosa
9          setosa
19         setosa
56     versicolor
104     virginica
69     versicolor
55     versicolor
132     virginica
29         setosa
127     virginica
26         setosa
128     virginica
131     virginica
145     virginica
108     virginica
143     virginica
45         setosa
30         setosa
Name: species, dtype: object
```

In [115]:

```
1 from sklearn.metrics import accuracy_score
2 accuracy_score(ytest,predictions)
```

Out[115]:

```
1.0
```

In [ ]:

```
1
```