PCAevaluacionAccuracy

September 14, 2019

In [1]:
```
#CARGANDO DATASET
import numpy as np
import pandas as pd
url = "https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data"
names = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'Class']
dataset = pd.read_csv(url, names=names)
dataset.head()
#print(dataset)
```

Out[1]:

	sepal-length	sepal-width	petal-length	petal-width	Class
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

In [4]:
```
#PREPROCESAMIENTO
X = dataset.drop('Class', 1)
y = dataset['Class']
```

In [6]:
```
# Splitting dataset en conjunto de training y conjunto de Test
from sklearn.model_selection import train_test_split

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

	sepal-length	sepal-width	petal-length	petal-width
114	5.8	2.8	5.1	2.4
62	6.0	2.2	4.0	1.0
33	5.5	4.2	1.4	0.2
107	7.3	2.9	6.3	1.8
7	5.0	3.4	1.5	0.2
100	6.3	3.3	6.0	2.5
40	5.0	3.5	1.3	0.3
86	6.7	3.1	4.7	1.5
76	6.8	2.8	4.8	1.4
71	6.1	2.8	4.0	1.3
134	6.1	2.6	5.6	1.4
51	6.4	3.2	4.5	1.5

73	6.1	2.8	4.7	1.2
54	6.5	2.8	4.6	1.5
63	6.1	2.9	4.7	1.4
37	4.9	3.1	1.5	0.1
78	6.0	2.9	4.5	1.5
90	5.5	2.6	4.4	1.2
45	4.8	3.0	1.4	0.3
16	5.4	3.9	1.3	0.4
121	5.6	2.8	4.9	2.0
66	5.6	3.0	4.5	1.5
24	4.8	3.4	1.9	0.2
8	4.4	2.9	1.4	0.2
126	6.2	2.8	4.8	1.8
22	4.6	3.6	1.0	0.2
44	5.1	3.8	1.9	0.4
97	6.2	2.9	4.3	1.3
93	5.0	2.3	3.3	1.0
26	5.0	3.4	1.6	0.4

In [4]:
```
#TRANSFORMACION MEDIANTE ESTANDARIZACION DE DATOS
from sklearn.preprocessing import StandardScaler

sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

In [20]:
```
#APLICACION DE PCA SIN ESPECIFICAR LAS CARACTERISTICAS
from sklearn.decomposition import PCA

pca = PCA()
#pca = PCA(n_components=2)
X_train = pca.fit_transform(X_train)
X_test = pca.transform(X_test)
explained_variance = pca.explained_variance_ratio_
print(explained_variance)
```

```
[0.72226528 0.23974795 0.03338117 0.0046056 ]
```

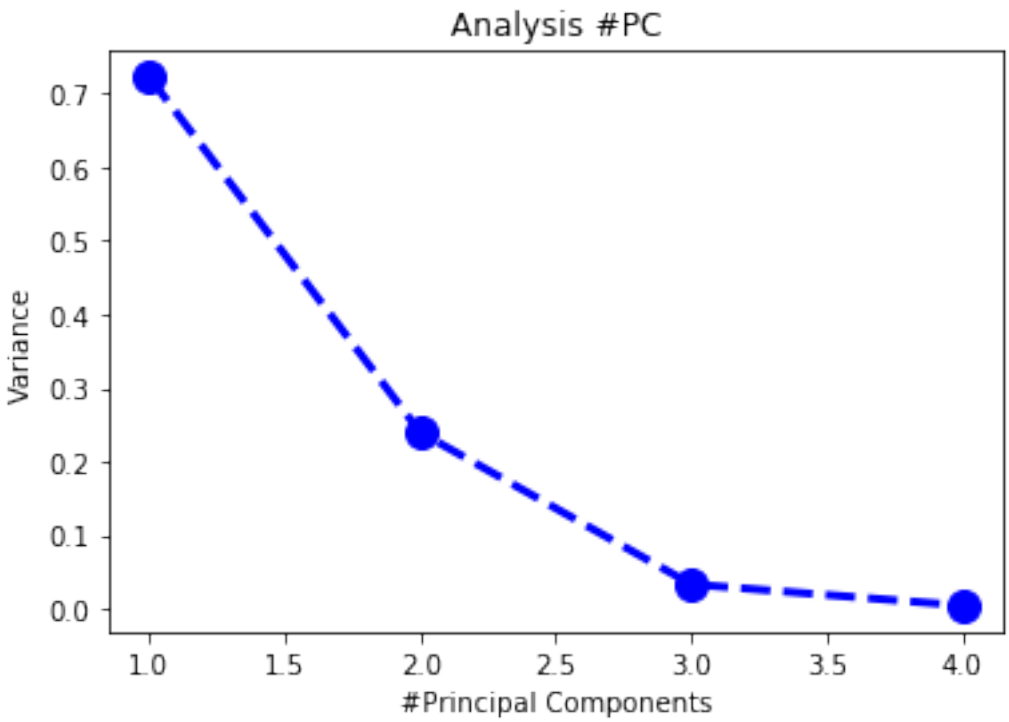
In [21]:
```
#Training and Making Predictions
#In this case we'll use random forest classification for making the predictions.
from sklearn.ensemble import RandomForestClassifier

classifier = RandomForestClassifier(max_depth=2, random_state=0)
classifier.fit(X_train, y_train)

# Predicting the Test set results
y_pred = classifier.predict(X_test)
```

```
C:\Users\UPS\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The de
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
```

In [22]:
```
#EVALUACION
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score

cm = confusion_matrix(y_test, y_pred)
print(cm)
print('accuracy: ' + str(accuracy_score(y_test, y_pred)))
```

```
[[11  0  0]
 [ 0 10  3]
 [ 0  1  5]]
accuracy: 0.8666666666666667
```

In [10]:
```
explained_variance
num_pc= [1,2,3,4]

#Visualize 2D Projection
import matplotlib.pyplot as plt

#plt.plot(num_pc, explained_variance)
plt.plot(num_pc, explained_variance, color='blue', linestyle='dashed', linewidth = 3,
         marker='o', markerfacecolor='blue', markersize=12)
plt.title('Analysis #PC')
plt.xlabel('#Principal Components')
plt.ylabel('Variance')
plt.show()
```



In [17]:
```
#PROCESO PCA CON N_COMPONENTS. LA MAYOR CANTIDAD DE COMPONENTES NO GARANTIZA UN MEJOR
from sklearn.model_selection import train_test_split

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

from sklearn.preprocessing import StandardScaler

sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

from sklearn.decomposition import PCA

#pca = PCA()
pca = PCA(n_components=1)
X_train = pca.fit_transform(X_train)
X_test = pca.transform(X_test)

explained_variance = pca.explained_variance_ratio_
print(explained_variance)

from sklearn.ensemble import RandomForestClassifier

classifier = RandomForestClassifier(max_depth=2, random_state=0)
```

```
classifier.fit(X_train, y_train)

# Predicting the Test set results
y_pred = classifier.predict(X_test)

from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score

cm = confusion_matrix(y_test, y_pred)
print(cm)
print('accuracy: ' + str(accuracy_score(y_test, y_pred)))
```

```
[0.72226528]
[[11  0  0]
 [ 0 12  1]
 [ 0  1  5]]
accuracy: 0.9333333333333333
```

```
C:\Users\UPS\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The de
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
```

In [18]:
```
#PROCESO SIN PCA

from sklearn.model_selection import train_test_split

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

from sklearn.preprocessing import StandardScaler

sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

from sklearn.ensemble import RandomForestClassifier

classifier = RandomForestClassifier(max_depth=2, random_state=0)
classifier.fit(X_train, y_train)

# Predicting the Test set results
y_pred = classifier.predict(X_test)

from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score

cm = confusion_matrix(y_test, y_pred)
print(cm)
print('accuracy: ' + str(accuracy_score(y_test, y_pred)))
```

```
[[11  0  0]
 [ 0 13  0]
 [ 0  2  4]]
accuracy: 0.9333333333333333


C:\Users\UPS\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The de
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
```

In [ ]: