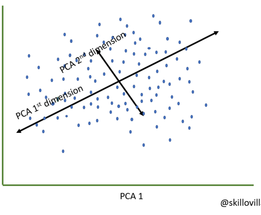
Predicting Heart Disease by using Principal Component Analysis (PCA)

- Large datasets are increasingly widespread in many disciplines. In order to interpret such datasets, methods are required to drastically reduce their dimensionality in an interpretable way, such that most of the information in the data is preserved. Many techniques have been developed for this purpose, but principal component analysis (PCA) is one of the oldest and most widely used. Its idea is simple—reduce the dimensionality of a dataset, while preserving as much 'variability' as possible.

Most datasets are heavy and dealing with all the features of a dataset can become unrealistic. The higher the number of features, the harder it is to classify the data-set. To add to that, some features are dependent and overlap with each other.

Our much needed fix, dimensionality reduction helps in reducing the number of features in an optimum manner such that we have lesser variables to work with. PCA allows this dimensionality reduction.

PCA generates a new set of features for our data-set which are called principal components. The features are arranged in descending order such that the feature with maximum variance to the original data is at the top.



- Why to use PCA

1. useful in processing data where multi-colinearity exists between the features/variables.
2. PCA can be used when the dimensions of the input features are high (e.g. a lot of variables).
3. PCA can be also used for denoising and data compression.

- The basic idea of PCA, leading to low-dimensional representations of large datasets in an adaptive and insightful way, is simple. there are many ways to adapt PCA to achieve modified goals or to analyse data of different types. Because PCA is used in a large number of areas, research into modifications and adaptations is spread over literatures from many disciplines.

Working of PCA

- PCA begins with the covariance (or correlation) matrix. First, we calculate the covariance of all the original variables and create the covariance matrix.
- For this covariance (or correlation) matrix, we now calculate the eigenvectors and eigenvalues.
- Every eigenvector would be a column vector with as many elements as the number of variables in the original dataset. Thus if we had an initial dataset of the size T x n (recall: rows are the observations, columns represent variables, therefore we have T observations of n variables), the covariance matrix would be of the size n x n, and each of the eigenvectors will be n x 1.
- The eigenvalues for each of the eigenvectors represent the amount of variance that the given eigenvector accounts for. We arrange the eigenvectors in decreasing order of the eigenvalues, and pick the top 2, 3 or as many eigenvalues that we are interested in depending upon how much variance we want to capture in our model. If we include all the eigenvectors, then we would have captured all the variance but this would not give us any advantage over our initial data.
- In a simplistic way, that is about all that there is to PCA. If you are reading the above for the first time, I would understand if it all gobbledygook. But no worries, we are going to go through an example to illustrate exactly how all of the above is done.

In [1]:
```
#importing pandas for data handling
import pandas as pd #for data manipulation
#for numerical operation
import numpy as np #for array operation
#importing matplot for basic plotting
import matplotlib.pyplot as plt #for 2-d plotting
#importing seaborn for advanced plotting
import seaborn as sns #it works on top of matplotlib
%matplotlib inline #magic function
#importing principal component analysis from sk.learn
from sklearn.decomposition import PCA
import umap
#importing magic functio for inline programme
%matplotlib inline
import os
```

About Dataset

The dataset contains the following features:

- age(in years)
- sex: (1 = male; 0 = female)
- cp: chest pain type
- trestbps: resting blood pressure (in mm Hg on admission to the hospital)
- chol: serum cholesterol in mg/dl
- fbs: (fasting blood sugar > 120 mg/dl) (1 = true; 0 = false)
- restecg: resting electrocardiographic results
- thalach: maximum heart rate achieved
- exang: exercise induced angina (1 = yes; 0 = no)
- oldpeak: ST depression induced by exercise relative to rest
- slope: the slope of the peak exercise ST segment
- ca: number of major vessels (0-3) colored by flourosopy
- thal: 3 = normal; 6 = fixed defect; 7 = reversable defect
- target: 1 or 0

In [2]:
```
#Reading the dataset
df = pd.read_csv("heart.csv")
df
```

Out[2]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
298	57	0	0	140	241	0	1	123	1	0.2	1	0	3	0
299	45	1	3	110	264	0	1	132	0	1.2	1	0	3	0
300	68	1	0	144	193	1	1	141	0	3.4	1	2	3	0
301	57	1	0	130	131	0	1	115	1	1.2	1	1	3	0
302	57	0	1	130	236	0	0	174	0	0.0	1	1	2	0

303 rows × 14 columns

In [3]:
```
#Checking missing values
df.isnull().sum()
```

Out[3]:
```
age         0
sex         0
cp          0
trestbps    0
chol        0
fbs         0
restecg     0
thalach     0
exang       0
oldpeak     0
slope       0
ca          0
thal        0
target      0
dtype: int64
```

In [4]:
```
feat=df.drop(['target'],axis=1)
```

In [5]:
```
target=df['target']
```

Correlation

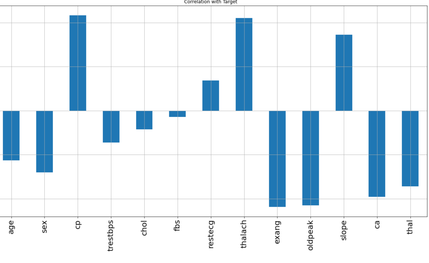
- Correlation is a statistical term describing the degree to which two variables move in coordination with one another. If the two variables move in the same direction, then those variables are said to have a positive correlation. If they move in opposite directions, then they have a negative correlation

- Correlation is a statistic that measures the degree to which two variables move in relation to each other.

- Correlation measures association, but doesn't show if x causes y or vice versa—or if the association is caused by a third factor

In [6]:
```
X=df.drop(['target'],axis=1)
X.corrwith(df['target']).plot.bar(
        figsize = (20, 10), title = "Correlation with Target", fontsize = 20,
        rot = 90, grid = True)
```

Out[6]:
```
<AxesSubplot:title={'center':'Correlation with Target'}>
```



PCA

- PCA is a technique which helps us in extracting a new set of variables from an existing large set of variables. These newly extracted variables are called Principal Components.

A principal component is a linear combination of the original variables Principal components are extracted in such a way that the first principal component explains maximum variance in the dataset Second principal component tries to explain the remaining variance in the dataset and is uncorrelated to the first principal component Third principal component tries to explain the variance which is not explained by the first two principal components and so on

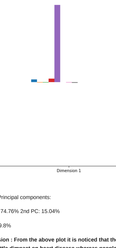
As the dataset is small having less features we will only 2 components or dimensions to see how much variance it is covering

- Applications of Principal Component Analysis (PCA)
- Principal Component Analysis can be used in Image compression. Image can be resized as per the requirement and patterns can be determined.
- Principal Component Analysis helps in Customer profiling based on demographics as well as their intellect in the purchase.
- PCA is a technique that is widely used by researchers in the food science field.
- It can also be used in the Banking field in many areas like applicants applied for loans, credit cards, etc.
- Customer Perception towards brands.
- It can also be used in the Finance field to analyze stocks quantitatively, forecasting portfolio returns, also in the interest rate implantation.
- PCA is also applied in Healthcare industries in multiple areas like patient insurance data where there are multiple sources of data and with a huge number of variables that are correlated to each other. Sources are like hospitals, pharmacies, etc.

In [7]:
```
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
pca_result = pca.fit_transform(feat.values)
```

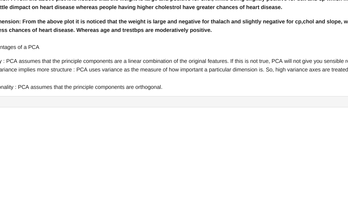
In [8]:
```
plt.plot(range(2), pca.explained_variance_ratio_)
plt.plot(range(2), np.cumsum(pca.explained_variance_ratio_))
plt.title("Component-wise and Cumulative Explained Variance")
```

Out[8]:
```
Text(0.5, 1.0, 'Component-wise and Cumulative Explained Variance')
```



- In the above graph, the blue line represents component-wise explained variance while the orange line represents the cumulative explained variance. We are able to explain around 90% variance in the dataset using just two components. Let us now try to visualize each of these decomposed components

In [15]:
```
def pca_results(good_data, pca):
    '''
    Create a DataFrame of the PCA results
    Includes dimension feature weights and explained variance
    Visualizes the PCA results
    '''

# Dimension indexing
    dimensions = dimensions = ['Dimension {}'.format(i) for i in range(1,len(pca.components_)+1)]

# PCA components
    components = pd.DataFrame(np.round(pca.components_, 4), columns = list(good_data.keys()))
    components.index = dimensions

    # PCA explained variance
    ratios = pca.explained_variance_ratio_.reshape(len(pca.components_), 1)
    variance_ratios = pd.DataFrame(np.round(ratios, 4), columns = ['Explained Variance'])
    variance_ratios.index = dimensions

# Create a bar plot visualization
    fig, ax = plt.subplots(figsize = (14,8))

# Plot the feature weights as a function of the components
    components.plot(ax = ax, kind = 'bar');
    ax.set_ylabel("Feature Weights")
    ax.set_xticklabels(dimensions, rotation=0)


# Display the explained variance ratios
    for i, ev in enumerate(pca.explained_variance_ratio_):
        ax.text(i-0.40, ax.get_ylim()[1] + 0.05, "Explained Variance\n          %.4f"%(ev))

# Return a concatenated DataFrame
    return pd.concat([variance_ratios, components], axis = 1)

pca_results = pca_results(feat, pca)
```



Inference

- First 2 Principal components:
- 1st PC: 74.76% 2nd PC: 15.04%
- Total: 89.8%

first dimension : From the above plot it is noticed that the weight is large and positive for chol, while being slightly positive for sex and cp which means that customers who score highly in this component will have very little dimpact on heart disease whereas people having higher cholestrol have greater chances of heart disease.

Second Dimension: From the above plot it is noticed that the weight is large and negative for thalach and slightly negative for cp,chol and slope, which means that patients who score high in this component will have very less chances of heart disease. Whereas age and trestbps are moderatively positive.

- disadvantages of a PCA

- Linearity : PCA assumes that the principle components are a linear combination of the original features. If this is not true, PCA will not give you sensible results.
- Large variance implies more structure : PCA uses variance as the measure of how important a particular dimension is. So, high variance axes are treated as principle components, while low variance axes are treated as noise.
- Orthogonality : PCA assumes that the principle components are orthogonal.

In [ ]:
```

```