Fourth Industrial Summer School

Day 2: Data Analysis Foundations – Afternoon

Feature Engineering

Dimensionality Reduction

Session Objectives

- ✓ Feature engineering
- ✓ Removing features with low variance
- ✓ Univariate Selection
- ✓ Recursive Feature Elimination
- ✓ Principal Component Analysis

Exploratory Data Analysis (EDA)

- Features selection
  - Low variance
  - Univariate analysis

- Dimensionality Reduction approaches
  - Principal Component Analysis (PCA)
  - Singular Value Decomposition (SVD)

- Both classical linear dimensionality reduction methods that attempt to find linear combinations of features in the original high dimensional data matrix to construct meaningful representation of the dataset.

Feature Engineering

- Feature engineering is a process of transforming the given data into a form which is easier to interpret.

- Tasks
  - Feature transformation
    - constructing new features from existing feature
  - Feature generation
    - generating new features that are often not the result of feature transformation.
  - Feature selection
    - selecting a small set of features from a very large pool of features.

Dimensionality Reduction

- Goal is to reduce the dimensions of a d-dimensional dataset by projecting it onto a (k)-dimensional subspace (where $k<d$)
  - while retaining most of the information.

- Question
  - what is the size of k that represents the data well?

Why dimensionality Reduction?

- Efficiency (storage and computation time)
  - Large number of features in the dataset is one of the factors that affect both the training time as well as accuracy of machine learning models.
- Remove noise and irrelevant information
- If we need to manually select some features to remove, what would be our strategy?
  - Remove features with least variance
  - Merging correlated variables
  - Extract the most important features

Feature Selection

- Feature selection is a process where you automatically select those features in your data that contribute most to the prediction variable.

- Benefits of performing feature selection:
  - **Reduces Overfitting**
  - **Improves Accuracy**
  - **Reduces Training Time**

Feature Selection

- **scikit-learn** provides a number of **automatic** feature selection techniques
  - **Removing features with low variance**
  - **Univariate Selection**
  - **Recursive Feature Elimination**
  - **Principal Component Analysis**

https://scikit-learn.org/stable/

Technique 1

Removing features with low variance

Removing features with low variance

- Simplest technique

- Remove features with low variance
  - **Calculate** variance for each feature
  - **Remove** features under a specified threshold
    - By default, it removes all zero-variance features (features that have the same value in all samples)

$$\text{Variance} = \sigma^2 = \frac{\Sigma(x_i - \mu)^2}{n}$$

Example

- Use **VarianceThreshold** from sklearn.feature_selection

```
1  from sklearn.feature_selection import VarianceThreshold
2
3  data = [[0, 2, 0, 3], [0, 1, 4, 3], [0, 1, 1, 3]]
4  print('Raw Data \n',data)
5
6  thresholder = VarianceThreshold(threshold=0.2)
7  new_data = thresholder.fit_transform(data)
8
9  print('Variances \n',thresholder.variances_)
10 print('New Data \n',new_data)
```

```
Raw Data
 [[0, 2, 0, 3], [0, 1, 4, 3], [0, 1, 1, 3]]
<class 'list'>
Variances
 [0.         0.22222222 2.88888889 0.        ]
New Data
 [[2 0]
 [1 4]
 [1 1]]
```

- How many features were removed?

Iris dataset

```
X = iris.data
Y = iris.target
```

```
from sklearn.feature_selection import VarianceThreshold

print('Original shape: ',X.shape)

thresholder = VarianceThreshold(threshold=.5)
X_new = thresholder.fit_transform(X)

print('Reduced shape: ',X_new.shape)
print("Feature variances ",thresholder.variances_)
```

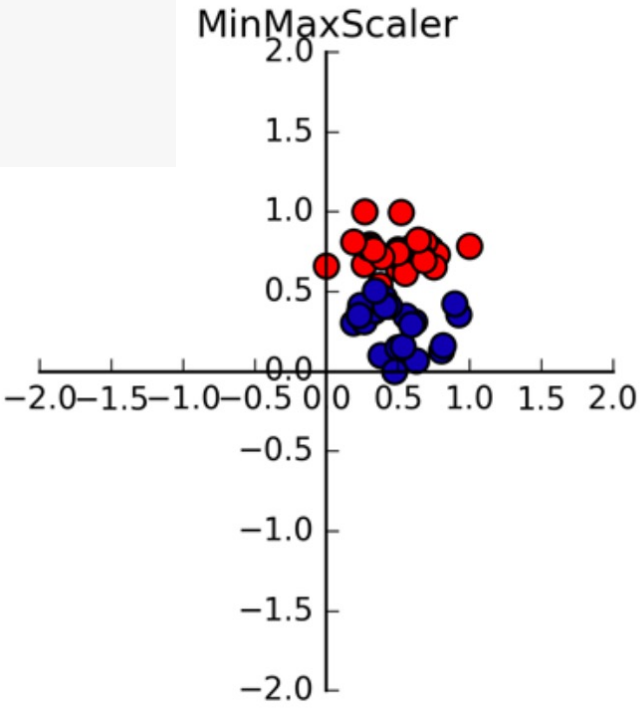
- Run the code, how many features were dropped? Which one?

Be careful

- Do we need to normalize/scale our data?

```
1  from sklearn.preprocessing import MinMaxScaler
2
3  scaler = MinMaxScaler()
4  scaler.fit(df)
5  scaled_features = scaler.transform(df)
6  print(scaled_features)
```

$$x_{\text{norm}} = \frac{x - \min(x)}{\max(x) - \min(x)}$$



Technique 2

Univariate Selection

Univariate Selection

- Univariate feature selection works by selecting the best features based on univariate statistical tests.
- Select those features that have the strongest relationship with the output variable.
- **SelectKBest** class provides a suite of different statistical tests to select a specific number of features.
- For **regression**: f_regression, mutual_info_regression
- For **classification**: chi2, f_classif, mutual_info_classif

Univariate Selection

- For **classification**
  - Chi2
    - Compute chi-squared stats between each non-negative feature and class.
  - f_classif
    - Compute the ANOVA F-value for the provided sample.
  - mutual_info_classif
    - Estimate mutual information for a discrete target variable.
- For **regression**
  - f_regression
    - Univariate linear regression tests returning F-statistic and p-values.
  - mutual_info_regression
    - Estimate mutual information for a continuous target variable.

Mutual information (MI) between two random variables is a non-negative value, which measures the dependency between the variables. It is equal to zero if and only if two random variables are independent, and higher values mean higher dependency.

Iris dataset

- Select best 2 features in the Iris dataset using the chi squared statistical test.
  - What is the highest two scores?

```
1  from sklearn.feature_selection import SelectKBest, chi2
2
3  fit = SelectKBest(chi2, k=2)
4  X_new = fit.fit_transform(X,Y)
5
6  print(X_new.shape)
7  print(fit.scores_)
8  print(fit.get_support(indices=True))
```

```
(150, 2)
[ 10.81782088   3.7107283  116.31261309  67.0483602 ]
[2 3]
```

Technique 3

Recursive Feature Elimination

Recursive Feature Elimination

- The Recursive Feature Elimination (RFE) works by recursively removing attributes and building a model on those attributes that remain.
  - assigns weights to features

- It uses the model **accuracy** to identify which attributes (and combination of attributes) contribute to predicting the target.

- Now, we will experiment with two models
  - Decision trees
  - K Nearest Neighbout
  - Logistic regression

Iris dataset

- Decision Trees

```
1  from sklearn.feature_selection import RFE
2  from sklearn.tree import DecisionTreeClassifier
3
4  model = DecisionTreeClassifier()
5  selector = RFE(model,n_features_to_select=2)
6  selector = selector.fit(X,Y)
7
8  print("Selected Features: ",selector.support_)
9  print("Feature Ranking: ",selector.ranking_)
```

```
Selected Features:  [False False  True  True]
Feature Ranking:  [3 2 1 1]
```

- What are the two features with highest ranking?

Iris dataset

- Try another Classifier and contrast features selected.

Technique 4

Principal Component Analysis (PCA)

Curse of Dimensionality

- Many Machine Learning problems involve thousands or even millions of features for each training instance.
- Slow training.
- Harder to find a good solution.

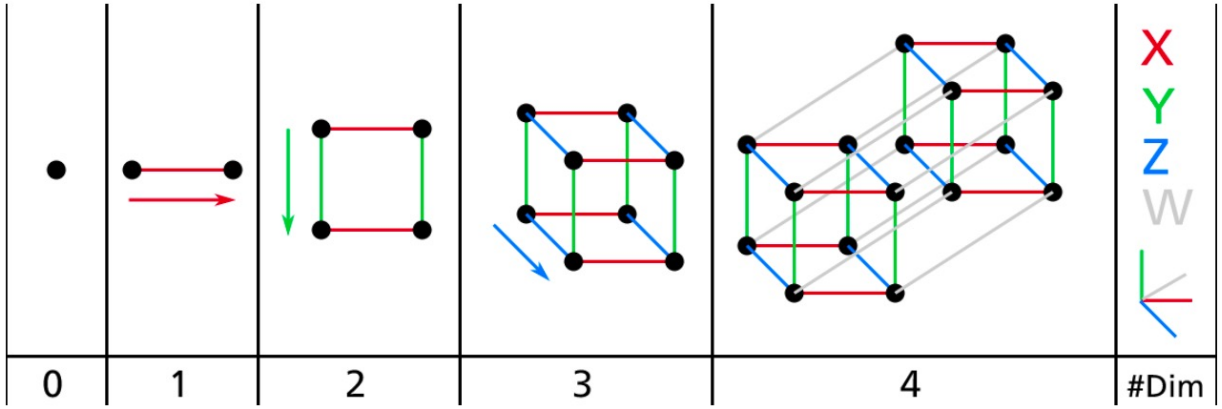


Figure 8-1. Point, segment, square, cube, and tesseract (0D to 4D hypercubes)[2]

Dimensionality Reduction

- Goal is to reduce the dimensions of a d-dimensional dataset by projecting it onto a (k)-dimensional subspace (where $k<d$)
  - while retaining most of the information.

- Question
  - what is the size of k that represents the data well?

Why dimensionality Reduction?

- Efficiency (storage and computation time)
  - Large number of features in the dataset is one of the factors that affect both the training time as well as accuracy of machine learning models.
- Remove noise and irrelevant information
- If we need to manually select some features to remove, what would be our strategy?
  - Remove features with least variance
  - Merging correlated variables
  - Extract the most important features

Principal Component Analysis (PCA)

- Principal component analysis (PCA), is an unsupervised learning technique to convert **high** dimensional data to **low** dimensional data by
  - Selecting the most important features that capture maximum information about the dataset.
  - PCA uses linear algebra to transform the dataset into a compressed form.

- Main goal of a PCA analysis is
  - Identify patterns in data
  - Detect the correlation between variables

- Detect the **correlation** between variables.
  - If a strong correlation between variables exists, we attempt to reduce the dimensionality

Principal Component Analysis (PCA)

- Feature selection is on the basis of variance that they cause in the output.
  - The feature that causes highest variance is the **first** principal component.
  - The feature that is responsible for **second** highest variance is considered the second principal component
  - so on …

- Advantages:
  - The **training time** of the algorithms reduces significantly with less number of features.
  - Difficult **analyze** data in high dimensions.
    - Plotting dataset with 200 features.

Principal Component Analysis (PCA)

- PCA is applied to **numeric** data.
  - Categorical features are required to be converted into numerical features before PCA can be applied.

https://www.analyticsvidhya.com/blog/2015/11/easy-methods-deal-categorical-variables-predictive-modeling/

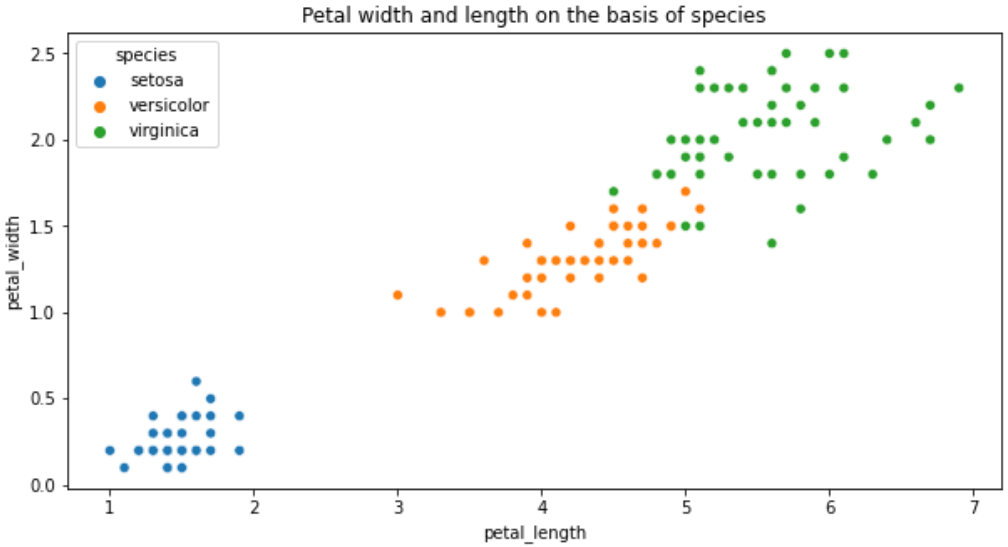
PCA Algorithm

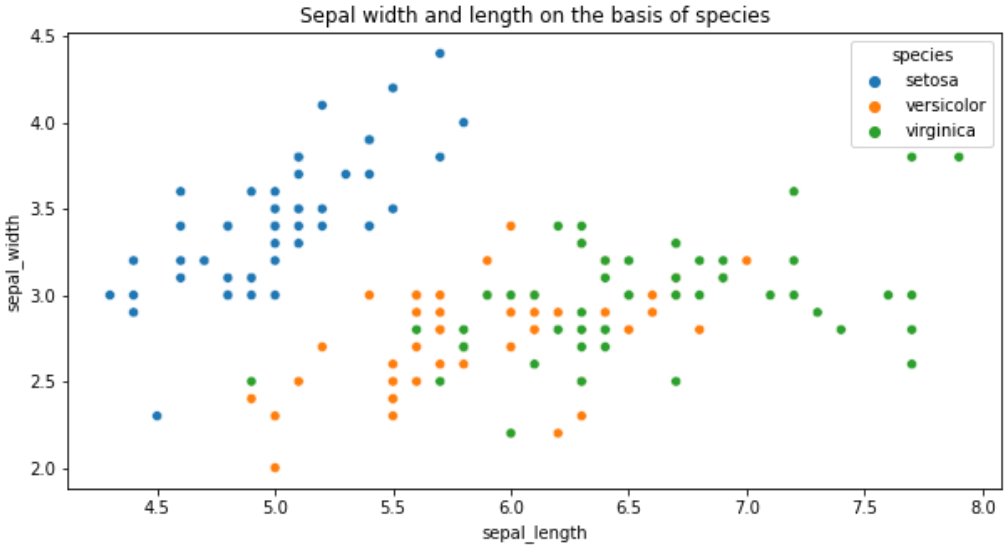
1. Standardize the ***d***-dimensional dataset
2. Calculate the covariance matrix
3. Decompose covariance matrix into its eigenvalues and eigenvectors
4. Sort the eigenvalues in descending order
   - Eigenvector with the highest value has the highest significance and forms the first principal component
5. Choose first ***k*** largest eigenvalues to be the new ***k*** dimensions.
6. Construct a projection matrix, ***W***, from the "top" ***k*** eigenvectors.
7. Transform the original ***d*** dimensional data points using ***W*** to obtain the new ***k*** dimensions.

Metaphor

Color in this metaphor is **variance** in the original data.

PCA - Iris dataset





PCA - Iris dataset (cont'd)

- Import needed libraries

```
1  # Necessary imports
2  from sklearn.decomposition import PCA
3  from sklearn.preprocessing import StandardScaler
4  from sklearn import datasets
5
6  # load data
7  iris = datasets.load_iris()
8  X = iris.data
9  Y = iris.target
```

PCA - Iris dataset (cont'd)

- Scale the features in your data before applying PCA.

- Use **StandardScaler** to help you standardize the dataset's features onto unit scale (mean = 0 and variance = 1)

```
1   X = StandardScaler().fit_transform(X)
```

- Feature scaling importance
  - Check link
  - https://scikit-learn.org/stable/auto_examples/preprocessing/plot_scaling_importance.html#sphx-glr-auto-examples-preprocessing-plot-scaling-importance-py

PCA - Iris dataset (cont'd)

- Features after scaling

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	-0.900681	1.019004	-1.340227	-1.315444
1	-1.143017	-0.131979	-1.340227	-1.315444
2	-1.385353	0.328414	-1.397064	-1.315444
3	-1.506521	0.098217	-1.283389	-1.315444

PCA - Iris dataset (cont'd)

- Project the 4-dimensional Iris data into 2

- Construct new dataframe from the PCs

```
1  pca = PCA(n_components=2)
2  principal_components = pca.fit_transform(X)
3  pc_df = pd.DataFrame(data=principal_components, columns=['PC1','PC2'])
4  pc_df.head(5)
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	-0.900681	1.019004	-1.340227	-1.315444
1	-1.143017	-0.131979	-1.340227	-1.315444
2	-1.385353	0.328414	-1.397064	-1.315444
3	-1.506521	0.098217	-1.283389	-1.315444

	PC1	PC2
0	-2.264703	0.480027
1	-2.080961	-0.674134
2	-2.364229	-0.341908
3	-2.299384	-0.597395
4	-2.389842	0.646835

PCA - Iris dataset (cont'd)

- Concatenate the target feature to our newly constructed dataframe which contains the PCs.

```
1   pc_df['target'] = Y
2   pc_df.head(5)
```

	PC1	PC2	target
0	-2.264703	0.480027	0
1	-2.080961	-0.674134	0
2	-2.364229	-0.341908	0
3	-2.299384	-0.597395	0
4	-2.389842	0.646835	0

PCA - Iris dataset (cont'd)

- How to make sure we haven't lost that much information?

- By using the attribute **explained_variance_ratio_**
  - First principal component contains **73%** of the variance
  - Second principal component contains **23%** of the variance.
  - Together, the two components contain **96%** of the information.

```
1   pca.explained_variance_ratio_
```

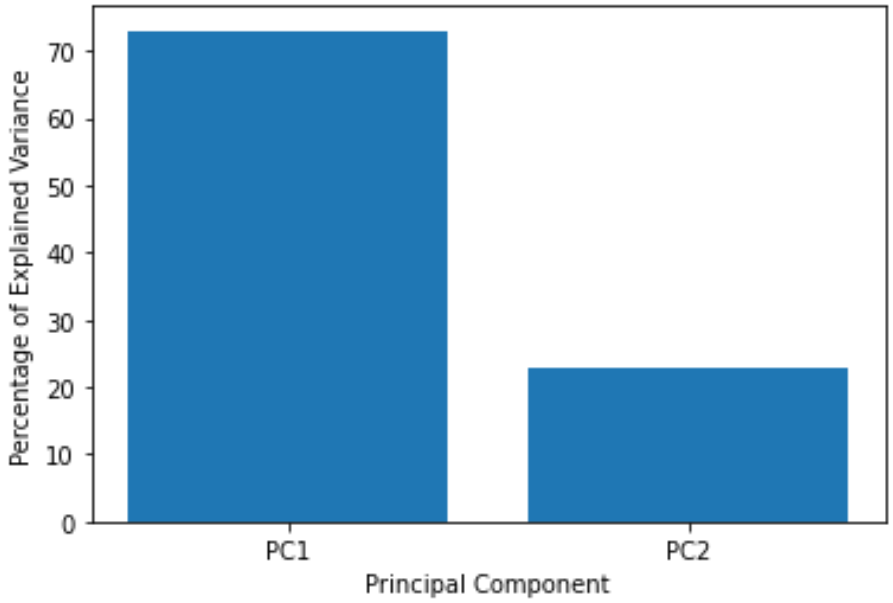
```
array([0.72962445, 0.22850762])
```

PCA - Iris dataset (cont'd)

- Plot PCs variance as a bar chart

```
per_var = np.round(pca.explained_variance_ratio_* 100)
labels  = ['PC' + str(x) for x in range(1, len(per_var)+1)]

plt.bar(x=range(1,len(per_var)+1), height=per_var, tick_label=labels)
plt.ylabel('Percentage of Explained Variance')
plt.xlabel('Principal Component')
plt.show()
```



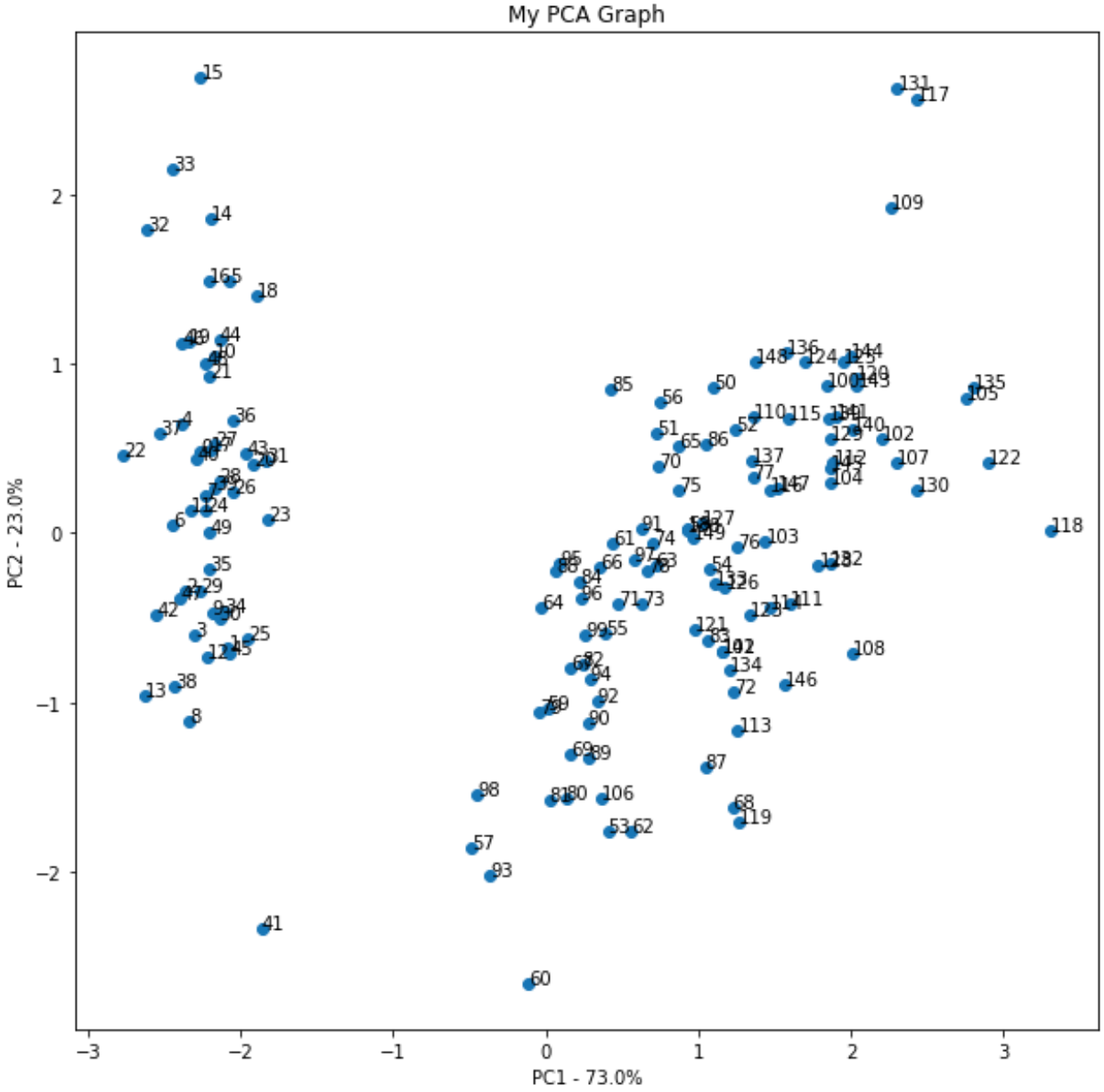
PCA - Iris dataset (cont'd)

- Plot the 2D data

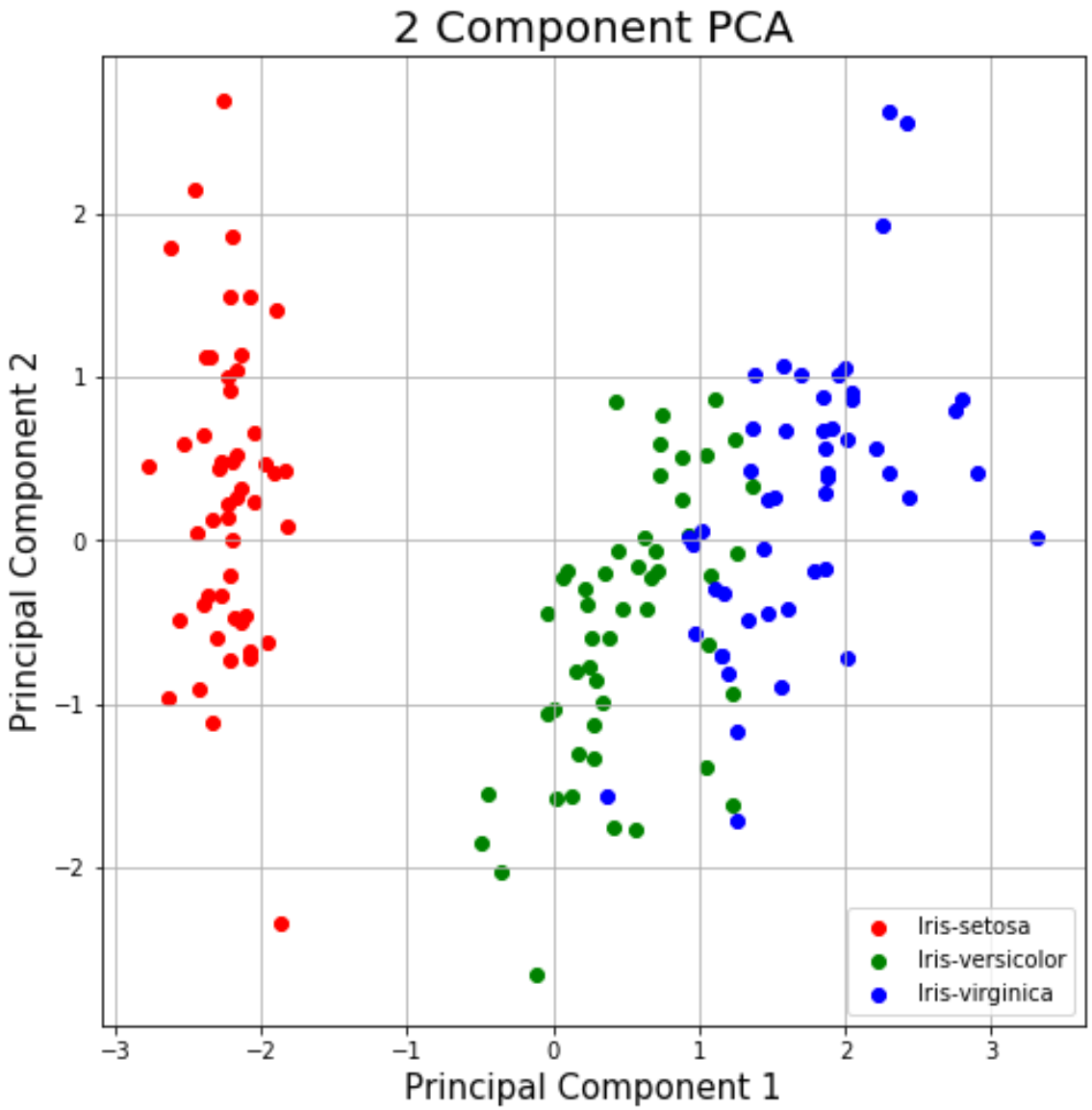
```
1   plt.figure(figsize = (10,10))
2   plt.scatter(principalDf.PC1, principalDf.PC2)
3   plt.title('My PCA Graph')
4   plt.xlabel('PC1 - {0}%'.format(per_var[0]))
5   plt.ylabel('PC2 - {0}%'.format(per_var[1]))
6
7   for sample in principalDf.index:
8       plt.annotate(sample, (principalDf.PC1.loc[sample], principalDf.PC2.loc[sample]))
9
10  plt.show()
```

Zip function: https://www.programiz.com/python-programming/methods/built-in/zip

- classes seem well separated from each other.

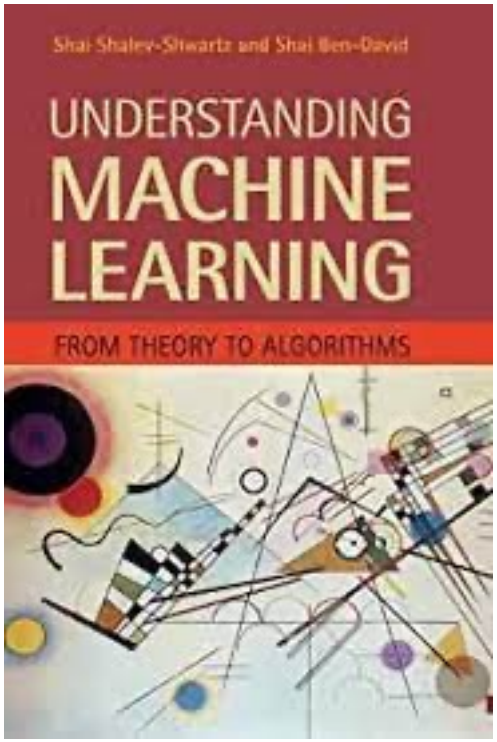


- classes seem well separated from each other.



Interested in more Math behind PCA.
Check this reference.

Chapter 23



Hands on session

Problem Solving

Exercise - Breast Cancer

- The Breast Cancer open source dataset
  - Target variable.
    - Two categories are: malignant and benign.
  - Total observations **569**
    - malignant class has 212 samples
    - benign class has 357 samples
  - **30** features
- Get it from either
  - sklearn
  - https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)

Exercise - Breast Cancer

- Perform the following feature selection on this dataset
  - **Removing features with low variance**
  - **Univariate Selection**
  - **Recursive Feature Elimination**
  - **Principal Component Analysis**

- **Share your solution with the instructor**