iCAN Lab KNU
Integrated Communications And Network Computing

Analysis of three ML algorithms for training MNIST dataset

Speaker: HeZhiming
SID: 2018227154
Date: 2019-12-06

Outlines

- Introduction to Three ML Algorithms (1 slide)
- Selected Platforms & Toolkits (2 slides)
- Project Results and Analysis (9 slides)
- Discussion and Summary (2 slides)

Introduction to Three ML Algorithms

- **Random Forest (Supervised Algorithms)**
  - constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees.

- **Support Vector Machine (Supervised Algorithms)**
  - given labeled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples.

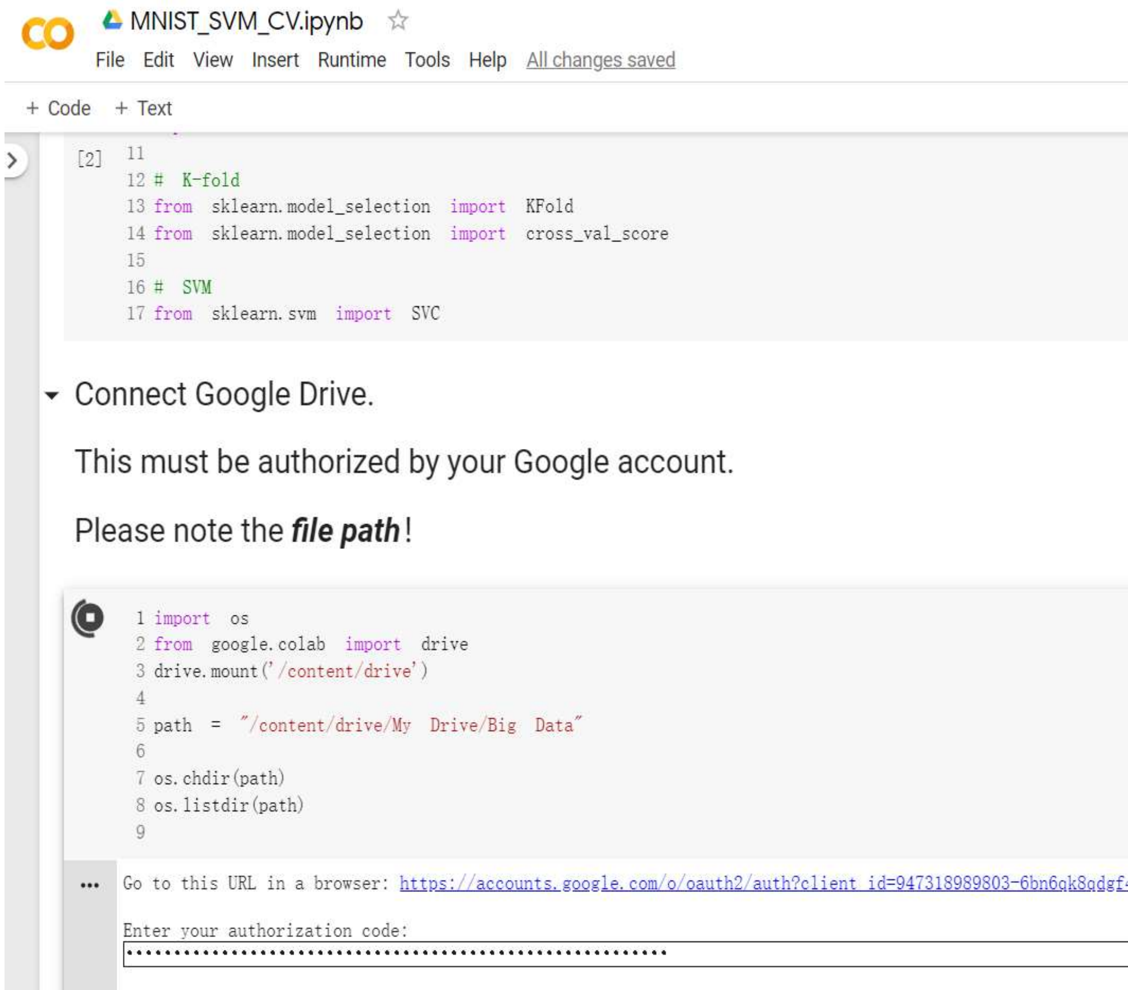
- **K-means (Unsupervised Algorithms)**
  - an iterative clustering analysis algorithm. Its steps are to randomly select k objects as the initial clustering center, then calculate the distance between each object and each seed clustering center and assign each object to the nearest clustering center.

◆ Outlines

- Introduction to Three ML Algorithms (1 slides)
- **Selected Platforms & Toolkits (2 slides)**
- Project Results and Analysis (9 slides)
- Discussion and Summary (2 slides)

Selected Platforms & Toolkits

- **Google Colaboratory (Colab)**
  1. Jupyter Notebook
  2. Can train ML models in the cloud
  3. Supports Python3 and Python2
  4. Provides **GPU** or **TPU** processor to quicken learning speed
  5. Source code and datasets are stored in Google Drive for easy access



Selected Platforms & Toolkits (cont.)

◆ Scikit-Learn (sklearn)

1. Well-known and open source python machine learning package
2. Simple and efficient tools for predictive data analysis
3. Accessible to everybody, and reusable in various contexts
4. Built on NumPy, SciPy, and matplotlib

import package

```
import pandas as pd
import numpy as np
# from sklearn import cross_validation
from sklearn.datasets import load_iris

# The function of train_test_split is Decompose t
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt
import csv

# K-fold
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score

# SVM
from sklearn.svm import SVC
```

iCAN Lab KNU Integrated Communications And Network Computing

Outlines

- Introduction to Three ML Algorithms (1 slides)
- Selected Platforms & Toolkits (2 slides)
- **Project Results and Analysis (9 slides)**
- Discussion and Summary (2 slides)

Project Results and Analysis – Random Forest

◆ Parameter Tuning for *max_depth*

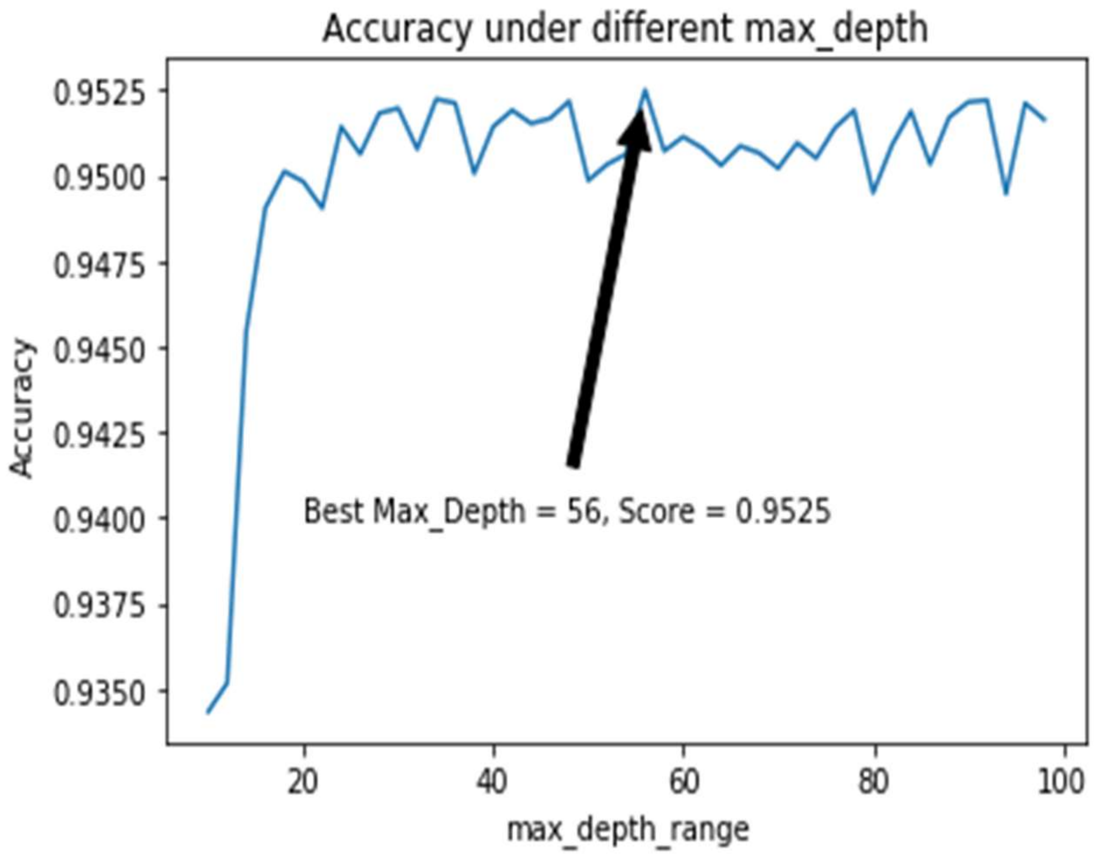
- (Built on **n_estimators = 20**)

1. **max_depth_range** = range(10,101,2)

2. Accuracy range: (0.9350, 0.9525]

3. When max_depth is **from 10 to 30**, the accuracy **increases greatly!**

4. When max_depth is **from 30 to 100**, the accuracy are **up and down motion, not the larger the better!**

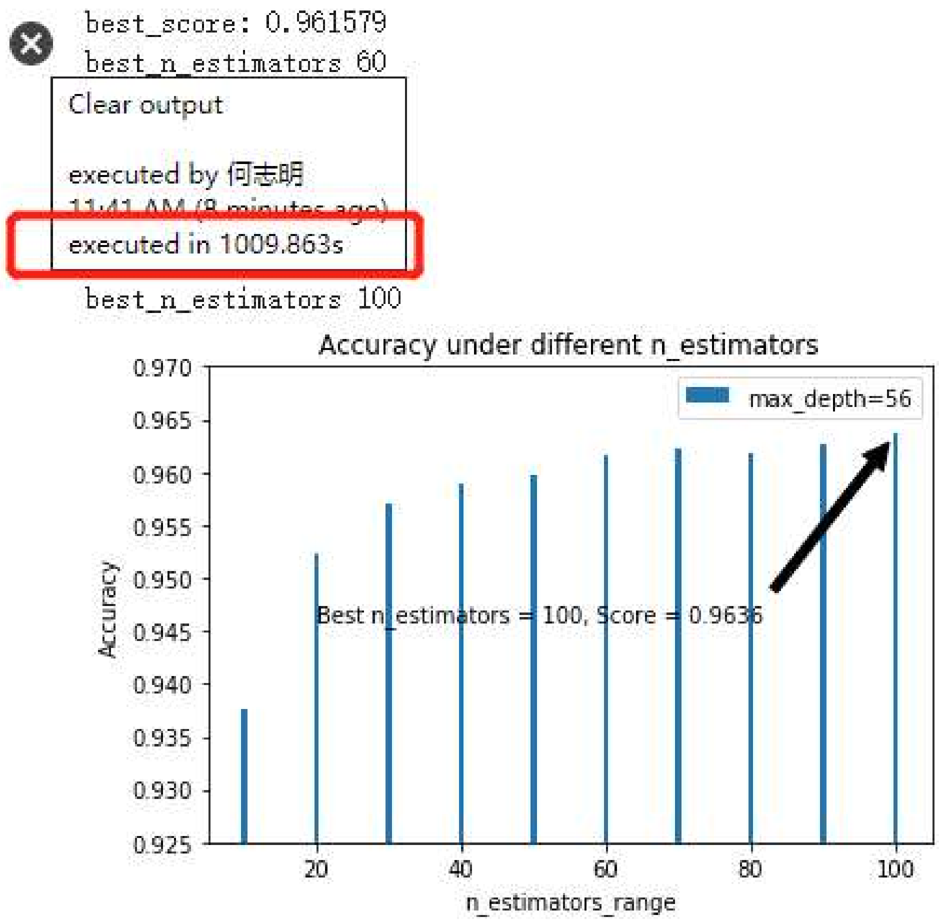


Project Results and Analysis – Random Forest (cont.)

◆ Parameter Tuning for *n_estimators*

- (Built on **max_depth = 56**)

1. **n_estimators_range**= range(10,101,10)
2. Accuracy range: (0.9357, 0.9637]
3. When n_estimators is from **10 to 100**, the accuracy increases.
4. In general, n_estimators are too small, and the model will be **underfitting**. **But** if n_estimators is too large, and the amount of calculation will be too **large**!

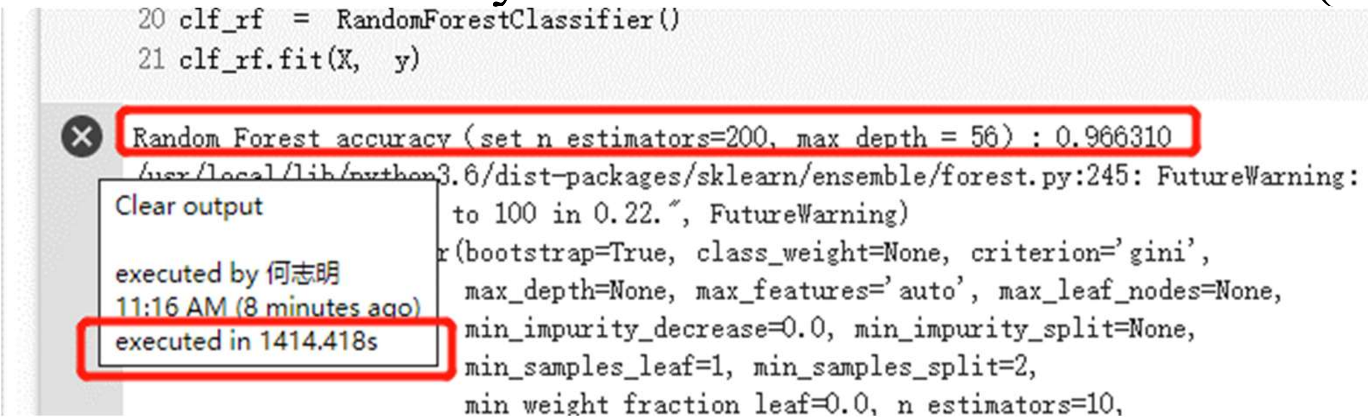


executed in 1009.863s

Project Results and Analysis – Random Forest (cont.)

◆ set n_estimators=200, max_depth = 56

1. The value of n_estimators has doubled
2. But the accuracy only increased by 0.02!
3. Execution time is nearly **half** of the last round of tests (**10 sets of results**)



executed in 1414.418s

iCAN Lab KNU
Integrated Communications And Network Computing

Project Results and Analysis – Support Vector Machine

◆ Parameter Tuning for *Penalty-Factor C*

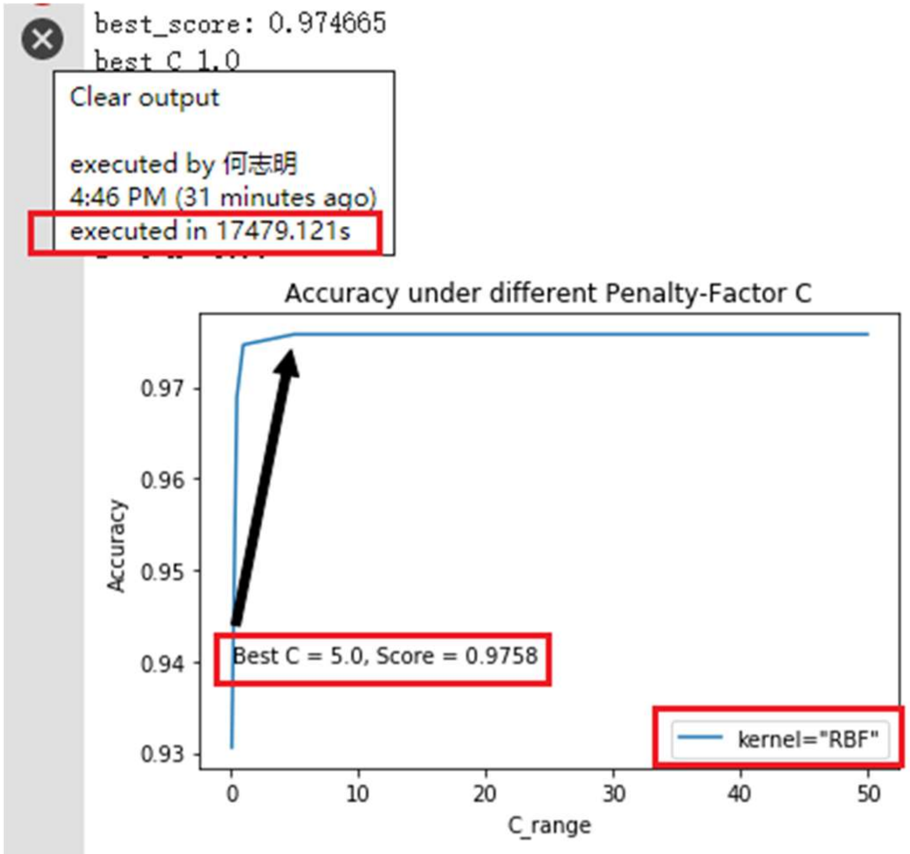
- (Built on **kernel = “RBF”**)

1. **C_range = np.array([0.1, 0.5, 1, 5, 10, 50])**

2. Accuracy range: (0.9306, 0.9758]

3. When C is from **0.1 to 5**, the accuracy increases. **But** after 5 (5, 10, 50), the accuracy has not changed.

4. Execution time is **TOO LONG!**



executed in 17479.121s

iCAN Lab KNU
Integrated Communications And Network Computing

Project Results and Analysis – Support Vector Machine (cont.)

◆ Parameter Tuning for *Kernel*

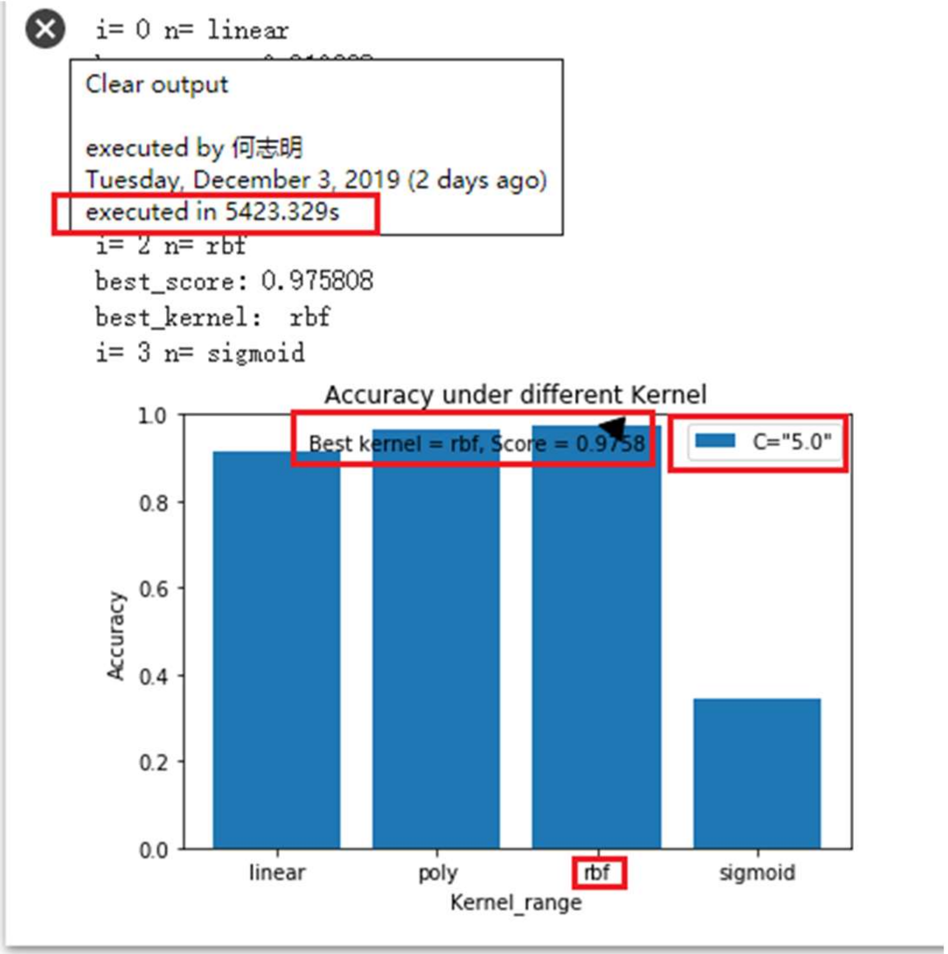
- (Built on **Penalty-Factor *C* = 5.0**)

1. **Kernel_range = np.array(['linear', 'poly', 'rbf ' , 'sigmoid'])**

1. Accuracy range: (0.327, 0.9758]

2. The **sigmoid** kernel function is **completely unsuitable** for MNIST data set, and its accuracy is **very low!**

3. The best kernel function is the **Radial Basis Function kernel.**



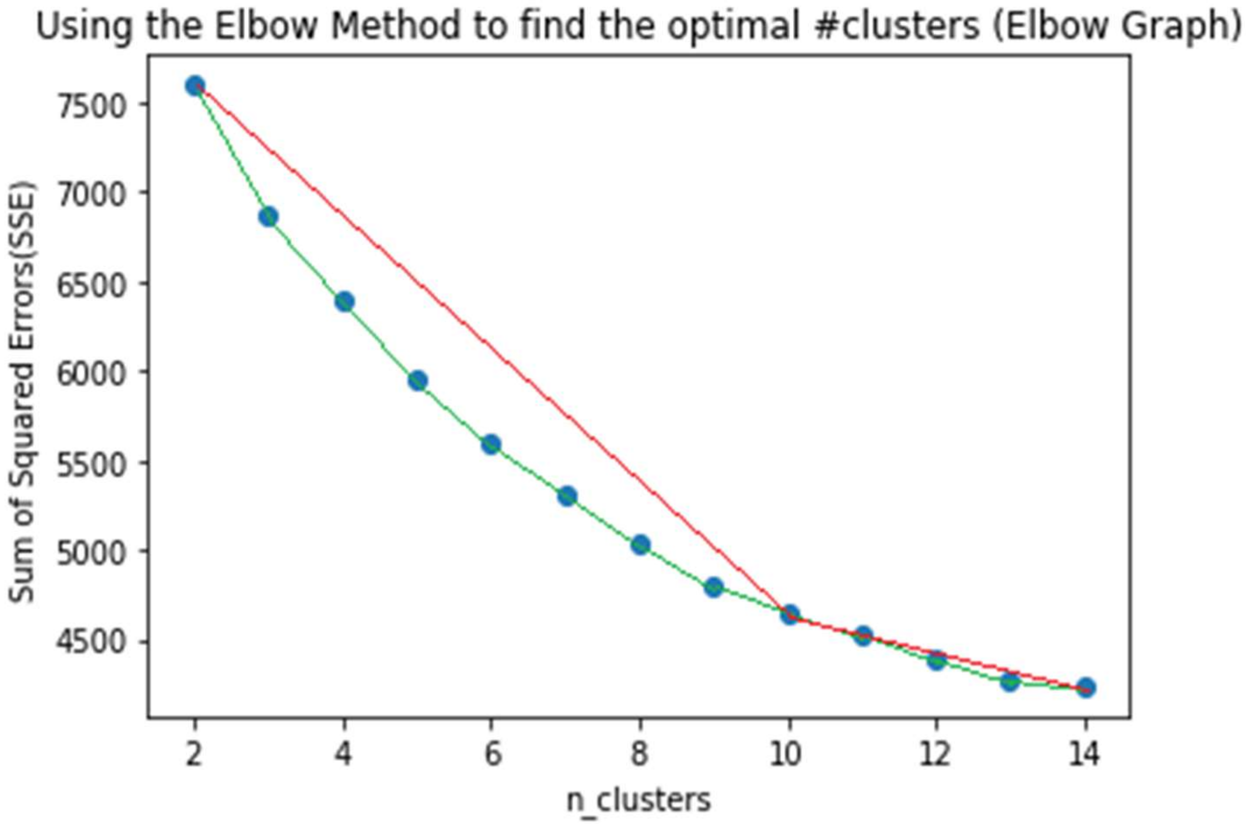
executed in 5423.329s (90 min)

Project Results and Analysis – K-Means

◆ Find the optimal # clusters

➢**K_Range = range (2, 15)**

➢From the reference line I added in this figure, it can be found that when **k = 10**, it is the **point of inflection.**



Project Results and Analysis – K-Means (cont.)

- **Applied t-SNE (Stochastic Neighbors Embedding) to reduce the dimensions of the dataset**

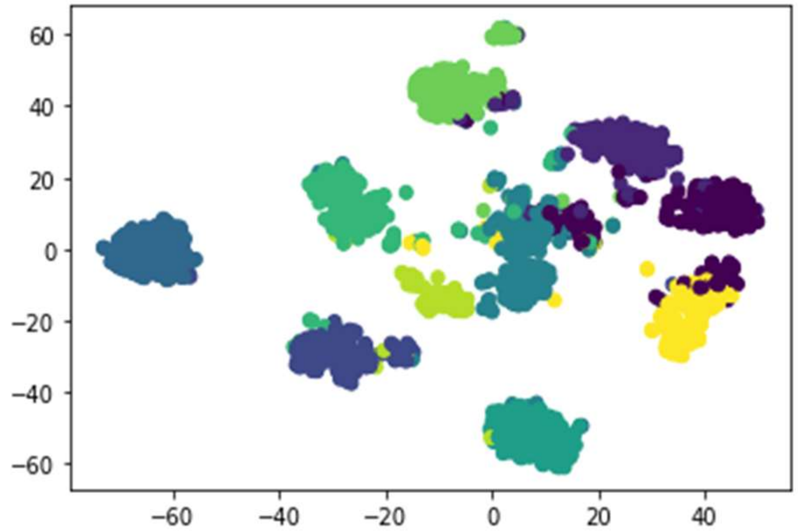
  - we should remember that k-means **is not a classification tool**, thus **analyzing accuracy** is not a very good idea. It is supposed to find a grouping of data which **maximizes between-clusters distances**, it does **not use labeling to train**.

```
[12]
from sklearn.manifold import TSNE

tsne=TSNE(2)
arr=tsne.fit_transform(df_img_scaled)
```

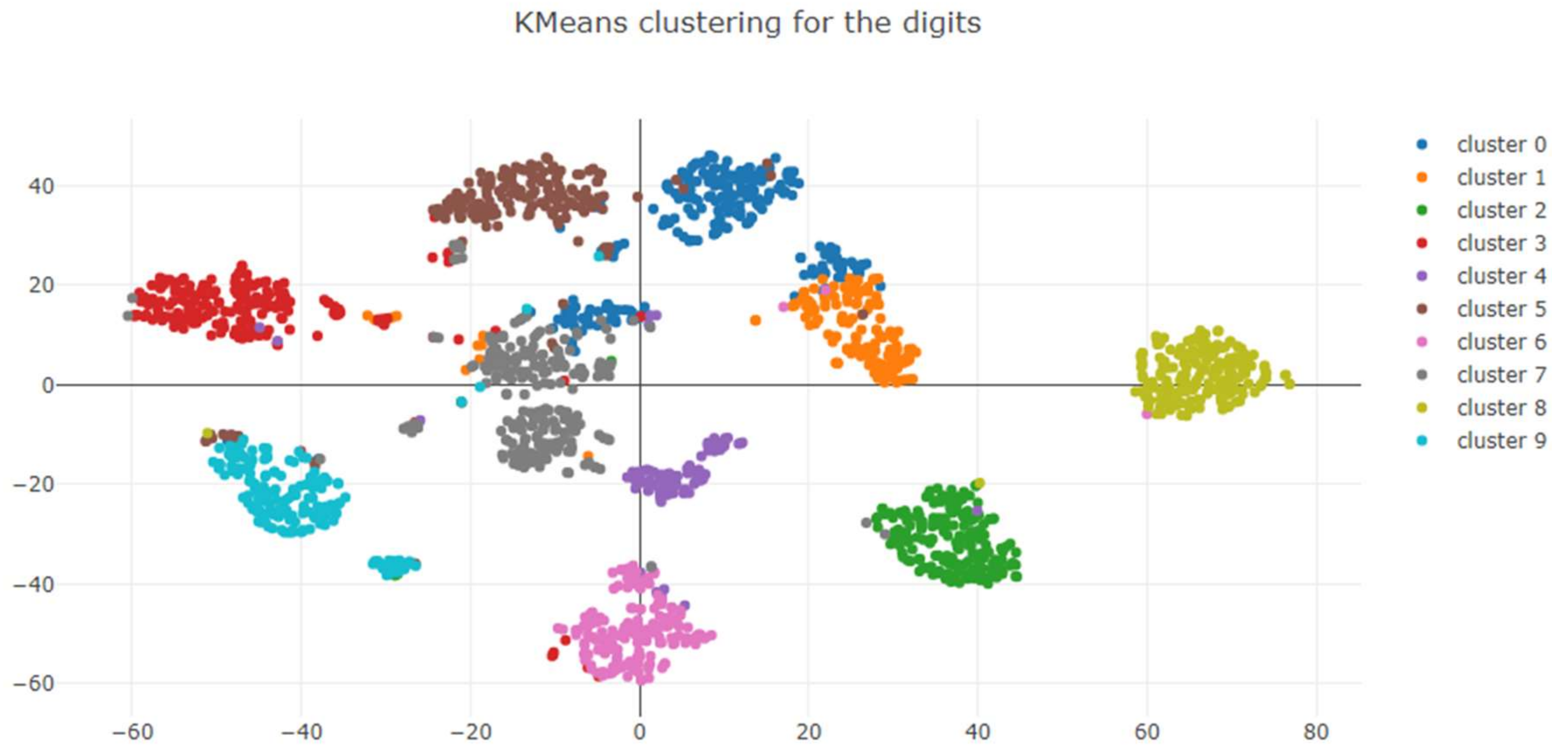
```
plt.scatter(arr[:,0],arr[:,1],c=model_10.labels_)
```

```
<matplotlib.collections.PathCollection at 0x7fc8adbc9f28>
```



Project Results and Analysis – K-Means (cont.)

- graph the resulting clusters in a 2D scatter plot using matplotlib and plotly



◆ Outlines

- Introduction to Three ML Algorithms (1 slides)
- Selected Platforms & Toolkits (2 slides)
- Project Results and Analysis (9 slides)
- **Discussion and Summary (2 slides)**

Discussion and Summary

- I can clearly feel that the running time of the **SVM** is very long compared to the other two algorithms, and it is estimated that it can reach a gap of a hundred times, but the accuracy of the SVM is really high. When the model is running, I specifically set it to reduce the running time. **test_size = 0.7** in the **train_test_split** function, but the accuracy can still reach **97.58%**, which is better than others.

- SO, Through this project, I summarized **three rules:**

1. **For Random Forest:** Able to train good results on large data sources **in a relatively short time.**

Discussion and Summary (cont.)

SO, Through this project, I summarized **three rules:**

2. **For Support Vector Machine:** When there are many training samples, the **efficiency** is not very high, but **accuracy** is better. And it is very important to **choose** the right kernel function (Accuracy of **sigmoid** is just 32.7%, accuracy of **rbf** is 97.6%).

3. **For K-Means:** Algorithm is very fast! The key point in this algorithm is to find the k partitions that **minimize the value of the squared error function.**

Q&As

QUESTIONS & ANSWERS

Thank You