T3 - Learning Curve

February 28, 2021

1 Learning Curve

1.1 Setting up

[ ]:
```
import pandas as pd
import numpy as np
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import Pipeline

# Load data
dataObj = load_breast_cancer()
X = dataObj.data
y = dataObj.target

# Splitting data
X_train, X_test, y_train, y_test = train_test_split(X, y,
    stratify=y,
    test_size=0.20,
    random_state=1)

# Constructing a pipeline object (without PCA)
pipe_lr = Pipeline([('scl', StandardScaler()),
                    ('clf', LogisticRegression(random_state=1,␣
 ↪max_iter=10000))])

# Constructing a pipeline object (with PCA)
#pipe_lr = Pipeline([('scl', StandardScaler()),
#                     ('pca', PCA(n_components=2)),
#                     ('clf', LogisticRegression(random_state=1,␣
 ↪max_iter=10000))])
```

[ ]:
```
# Different percentage of data used to construct a learning curve
train_size = np.linspace(0.1, 1.0, 10)
print(train_size)
```

```
[ ]: from sklearn.model_selection import learning_curve

     train_sizes, train_scores, val_scores =\
                     learning_curve(estimator=pipe_lr,
                                    X=X_train,
                                    y=y_train,
                                    train_sizes=train_size,
                                    cv=10,
                                    n_jobs=1)
```

```
[ ]: # Number of training samples
     print(train_sizes)
```

```
[ ]: # Training accuracy
     df = pd.DataFrame(train_scores)
     df.insert(loc=0, column="n_samples", value=train_sizes)
     df = df.set_index("n_samples")
     display(df)
```

```
[ ]: # Validation accuracy
     df = pd.DataFrame(val_scores)
     df.insert(loc=0, column="n_samples", value=train_sizes)
     df = df.set_index("n_samples")
     display(df)
```

```
[ ]: train_mean = np.mean(train_scores, axis=1)
     train_std = np.std(train_scores, axis=1)
     val_mean = np.mean(val_scores, axis=1)
     val_std = np.std(val_scores, axis=1)
```

```
[ ]: df = pd.DataFrame( \
         data=np.stack((train_sizes, train_mean, train_std, val_mean, val_std),␣
      ↪axis=1),
         columns=['n_samples','train_mean','train_std','val_mean','val_std'])
     df = df.set_index('n_samples')
     display(df)
```

```
[ ]: import matplotlib.pyplot as plt

     plt.plot(train_sizes, train_mean,
              color='blue', marker='o',
              markersize=5, label='training accuracy')

     plt.fill_between(train_sizes,
                      train_mean + train_std,
                      train_mean - train_std,
                      alpha=0.15, color='blue')
```

```
plt.plot(train_sizes, val_mean,
         color='green', linestyle='--',
         marker='s', markersize=5,
         label='validation accuracy')

plt.fill_between(train_sizes,
                 val_mean + val_std,
                 val_mean - val_std,
                 alpha=0.15, color='green')

plt.grid()
plt.xlabel('Number of training samples')
plt.ylabel('Accuracy')
plt.legend(loc='lower right')
plt.ylim([0.8, 1.0])
plt.tight_layout()
plt.show()
```