NAME : Darshan CD

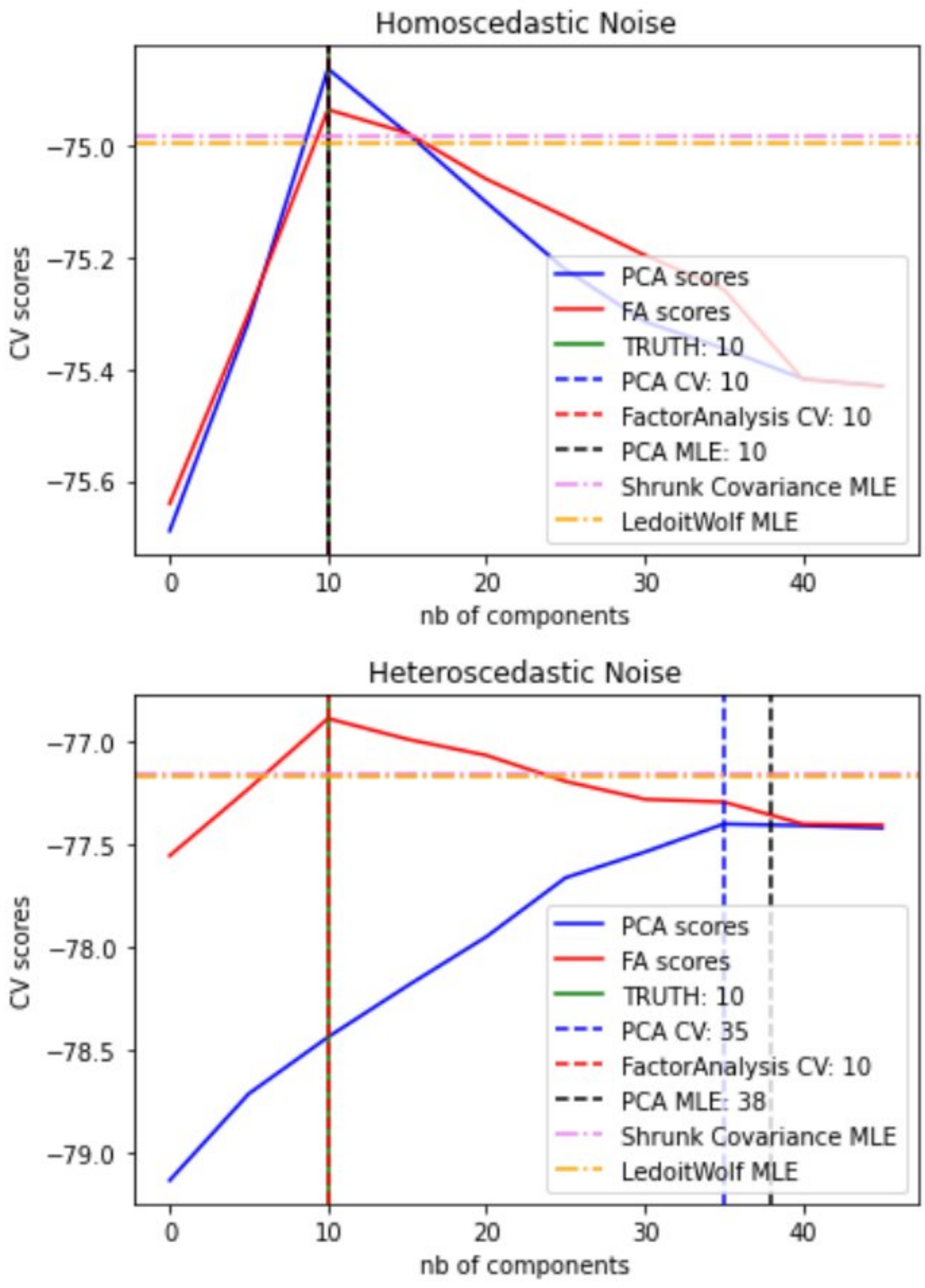
SECTION : 5CSE02

ROLL NUMBER: 20191CSE0115

```
import numpy as np
import matplotlib.pyplot as plt
from scipy importlinalg

from sklearn.decomposition import PCA, FactorAnalysis
from sklearn.covariance import ShrunkCovariance,
LedoitWolffrom sklearn.model_selection import cross_val_score
from sklearn.model_selection import GridSearchCV

print(doc)

# ##############################################################################
# Create the data

n_samples,n_features,rank=1000,50,10
sigma=1.
rng = np.random.RandomState(42)
U, _, _ = linalg.svd(rng.randn(n_features, n_features))
X = np.dot(rng.randn(n_samples, rank), U[:,:rank].T)

# Adding homoscedastic noise
X_homo = X + sigma * rng.randn(n_samples, n_features)

# Adding heteroscedastic noise
sigmas=sigma*rng.rand(n_features)+sigma/2.
X_hetero = X + rng.randn(n_samples, n_features) *sigmas

# ##############################################################################
# Fit the models

n_components=np.arange(0,n_features,5)        #optionsforn_components


defcompute_scores(X):
    pca = PCA(svd_solver='full')
    fa =FactorAnalysis()

    pca_scores, fa_scores = [], []
    for n in n_components:
        pca.n_components = n
        fa.n_components =n
        pca_scores.append(np.mean(cross_val_score(pca, X)))
        fa_scores.append(np.mean(cross_val_score(fa, X)))

    return pca_scores, fa_scores
```

```
defshrunk_cov_score(X):
    shrinkages=np.logspace(-2,0,30)
    cv = GridSearchCV(ShrunkCovariance(), {'shrinkage': shrinkages})
    return np.mean(cross_val_score(cv.fit(X).best_estimator_, X))


deflw_score(X):
    return np.mean(cross_val_score(LedoitWolf(), X))


for X, title in [(X_homo, 'Homoscedastic Noise'),
                 (X_hetero, 'Heteroscedastic Noise')]:
    pca_scores, fa_scores = compute_scores(X)
    n_components_pca = n_components[np.argmax(pca_scores)]
    n_components_fa = n_components[np.argmax(fa_scores)]

    pca = PCA(svd_solver='full', n_components='mle')
    pca.fit(X)
    n_components_pca_mle =pca.n_components_

    print("best n_components by PCA CV = %d" % n_components_pca)
    print("best n_components by FactorAnalysis CV = %d" % n_components_fa)
    print("best n_components by PCA MLE = %d" % n_components_pca_mle)

    plt.figure()
    plt.plot(n_components, pca_scores, 'b', label='PCA scores')
    plt.plot(n_components, fa_scores, 'r',label='FA scores')
    plt.axvline(rank, color='g', label='TRUTH: %d' % rank, linestyle='-')
    plt.axvline(n_components_pca, color='b',
                label='PCA CV: %d' % n_components_pca, linestyle='--')
    plt.axvline(n_components_fa,color='r',
                label='FactorAnalysis CV: %d' % n_components_fa,
                linestyle='--')
    plt.axvline(n_components_pca_mle, color='k',
                label='PCA MLE: %d' % n_components_pca_mle, linestyle='--')

    # compare with other covariance estimators
    plt.axhline(shrunk_cov_score(X), color='violet',
                label='ShrunkCovarianceMLE',linestyle='-.')
    plt.axhline(lw_score(X), color='orange',
                label='LedoitWolf MLE' % n_components_pca_mle, linestyle='-.')

    plt.xlabel('nb of components')
    plt.ylabel('CV scores')
    plt.legend(loc='lower right')
    plt.title(title)

plt.show()
```

```
Automatically created module for IPython interactive  environment
bestn_componentsbyPCACV=10
best n_components by FactorAnalysis CV  =  10
best n_components by PCA MLE =10
best n_components by PCA CV = 35
best n_components by FactorAnalysis CV  =  10
best n_components by PCA MLE =38
```



31s completed at 12:32PM