Module 18 Challenge

New Attempt

Due Monday by 12:59am **Points** 100 **Submitting** a text entry box or a website url

Background

You and Martha have done your research. You understand what unsupervised learning is used for, how to process data, how to cluster, how to reduce your dimensions, and how to reduce the principal components using PCA. It's time to put all these skills to use by creating an analysis for your clients who are preparing to get into the cryptocurrency market.

Martha is a senior manager for the Advisory Services Team at Accountability Accounting, one of your most important clients. Accountability Accounting, a prominent investment bank, is interested in offering a new cryptocurrency investment portfolio for its customers. The company, however, is lost in the vast universe of cryptocurrencies. So, they've asked you to create a report that includes what cryptocurrencies are on the trading market and how they could be grouped to create a classification system for this new investment.

The data Martha will be working with is not ideal, so it will need to be processed to fit the machine learning models. Since there is no known output for what Martha is looking for, she has decided to use unsupervised learning. To group the cryptocurrencies, Martha decided on a clustering algorithm. She'll use data visualizations to share her findings with the board.

What You're Creating

This new assignment consists of four technical analysis deliverables. You will submit the following:

- Deliverable 1: Preprocessing the Data for PCA
- Deliverable 2: Reducing Data Dimensions Using PCA
- Deliverable 3: Clustering Cryptocurrencies Using K-means
- Deliverable 4: Visualizing Cryptocurrencies Results

Files

Use the following links to download the dataset and Challenge starter code.

Download cryptocurrency data (crypto_data.csv) (https://2u-data-curriculum-team.s3.amazonaws.com/dataviz-online/module_18/crypto_data.csv)

Download crypto clustering starter code (https://2u-data-curriculum-team.s3.amazonaws.com/dataviz-online/module_18/crypto_clustering_starter_code.ipynb)

Deliverable 1: Preprocessing the Data for PCA (30 points)

Deliverable 1 Instructions

Using your knowledge of Pandas, you'll preprocess the dataset in order to perform PCA in Deliverable 2.

REWIND

For this deliverable, you've already done the following in this module:

- **Lesson 18.2.1:** Review the steps to prepare data
- **Lesson 18.2.2:** Pandas refresher
- **Lesson 18.2.3:** Preprocessing data with Pandas
- **Lesson 18.2.4:** Data selection
- **Lesson 18.2.5:** Preprocessing data
- **Lesson 18.5.2:** Use the StandardScaler library to standardize features

Follow the instructions below and use the `crypto_clustering_starter_code.ipynb` file to complete Deliverable 1.

Open the `crypto_clustering_starter_code.ipynb` file, rename it `crypto_clustering.ipynb`, and save it to your Cryptocurrencies GitHub folder.

1. Read in the `crypto_data.csv` to the Pandas DataFrame named `crypto_df`.

NOTE

The `crypto_data.csv` was retrieved from **CryptoCompare** (https://min-api.cryptocompare.com/data/all/coinlist).

2. Keep all the cryptocurrencies that are being traded.
3. Drop the `IsTrading` column.
4. Remove rows that have at least one null value.
5. Filter the `crypto_df` DataFrame so it only has rows where coins have been mined.
6. Create a new DataFrame that holds only the cryptocurrency names, and use the `crypto_df` DataFrame index as the index for this new DataFrame.
7. Remove the `CoinName` column from the `crypto_df` DataFrame since it's not going to be used on the clustering algorithm.

Take a moment to check that your `crypto_df` DataFrame looks like the image below:

	Algorithm	ProofType	TotalCoinsMined	TotalCoinSupply
42	Scrypt	PoW/PoS	4.199995e+01	42
404	Scrypt	PoW/PoS	1.055185e+09	532000000
1337	X13	PoW/PoS	2.927942e+10	314159265359
BTC	SHA-256	PoW	1.792718e+07	21000000
ETH	Ethash	PoW	1.076842e+08	0
LTC	Scrypt	PoW	6.303924e+07	84000000
DASH	X11	PoW/PoS	9.031294e+06	22000000
XMR	CryptoNight-V7	PoW	1.720114e+07	0
ETC	Ethash	PoW	1.133597e+08	210000000
ZEC	Equihash	PoW	7.383056e+06	21000000

8. Use the `get_dummies()` method to create variables for the two text features, `Algorithm` and `ProofType`, and store the resulting data in a new DataFrame named `X`.
9. Use the StandardScaler `fit_transform()` function to standardize the features from the `X` DataFrame.

IMPORTANT

Using the `StandardScaler()` sklearn library to standardize the features is required before attempting Deliverables 2 and 3.

Save your `crypto_clustering.ipynb` file to your Cryptocurrencies folder.

Deliverable 1 Requirements

You will earn a perfect score for Deliverable 1 by completing all requirements below:

- The following five preprocessing steps have been performed on the `crypto_df` DataFrame:
  - All cryptocurrencies that are not being traded are removed **(3 pt)**
  - The `IsTrading` column is dropped **(3 pt)**
  - All the rows that have at least one null value are removed **(3 pt)**
  - All the rows that do not have coins being mined are removed **(3 pt)**
  - The `CoinName` column is dropped **(3 pt)**

- A new DataFrame is created that stores all cryptocurrency names from the `CoinName` column and retains the index from the `crypto_df` DataFrame **(5 pt)**

- The `get_dummies()` method is used to create variables for the text features, which are then stored in a new DataFrame, `X` **(5 pt)**
- The features from the `X` DataFrame have been standardized using the StandardScaler `fit_transform()` function **(5 pt)**

Deliverable 2: Reducing Data Dimensions Using PCA (20 points)

Deliverable 2 Instructions

Using your knowledge of how to apply the Principal Component Analysis (PCA) algorithm, you'll reduce the dimensions of the `X` DataFrame to three principal components and place these dimensions in a new DataFrame.

REWIND

For this deliverable, you've already done the following in this module:

- **Lesson 18.5.2:** Apply Principal Component Analysis

Follow the instructions below and use the information in the `crypto_clustering_starter_code.ipynb` file to complete Deliverable 2.

1. Continue using the `crypto_clustering.ipynb` file from Deliverable 1 where you've already performed the preprocessing steps.
2. Using the information we've provided, apply PCA to reduce the dimensions to three principal components.

If you'd like a hint on how to use the PCA algorithm, that's totally okay. If not, that's great too. You can always revisit this later if you change your mind.

SHOW HINT

3. Create a new DataFrame named `pcs_df` that includes the following columns, `PC 1`, `PC 2`, and `PC 3`, and uses the index of the `crypto_df` DataFrame as the index.

Your DataFrame should look like the image below:

	PC 1	PC 2	PC 3
42	-0.335587	1.060721	-0.442783
404	-0.319468	1.060956	-0.443089
1337	2.298643	1.616176	-0.543968
BTC	-0.146783	-1.352007	0.158842
ETH	-0.134787	-1.929485	0.248370
LTC	-0.157018	-1.083072	0.028331
DASH	-0.409440	1.226179	-0.562194
XMR	-0.165529	-2.462011	0.404730
ETC	-0.133224	-1.929573	0.248345
ZEC	-0.128082	-1.987514	0.351649

Save your `crypto_clustering.ipynb` file to your Cryptocurrencies folder.

Deliverable 2 Requirements

You will earn a perfect score for Deliverable 2 by completing all requirements below:

- The PCA algorithm reduces the dimensions of the `X` DataFrame down to three principal components **(10 pt)**
- The `pcs_df` DataFrame is created and has the following three columns, `PC 1`, `PC 2`, and `PC 3`, and has the index from the `crypto_df` DataFrame **(10 pt)**

Deliverable 3: Clustering Cryptocurrencies Using K-means (20 points)

Deliverable 3 Instructions

Using your knowledge of the K-means algorithm, you'll create an elbow curve using `hvPlot` to find the best value for K from the `pcs_df` DataFrame created in Deliverable 2. Then, you'll run the K-means algorithm to predict the K clusters for the cryptocurrencies' data.

REWIND

For this deliverable, you've already done the following in this module:

- **Lesson 18.3.2:** Create an instance of the K-means algorithm and make predictions
- **Lesson 18.4.2:** Create an elbow curve using `hvPlot`
- **Lesson 18.5.2:** Apply Principal Component Analysis

Follow the instructions below and use the information in the `crypto_clustering_starter_code.ipynb` file to complete Deliverable 3.

1. Continue using the `crypto_clustering.ipynb` file that you used in Deliverable 2 to reduce the dataset to three dimensions.
2. Using the `pcs_df` DataFrame, create an elbow curve using `hvPlot` to find the best value for K.
3. Next, use the `pcs_df` DataFrame to run the K-means algorithm to make predictions of the K clusters for the cryptocurrencies' data.

If you'd like a hint on how to use the K-means algorithm, that's totally okay. If not, that's great too. You can always revisit this later if you change your mind.

SHOW HINT

4. Create a new DataFrame named `clustered_df` by concatenating the `crypto_df` and `pcs_df` DataFrames on the same columns. The index should be the same as the `crypto_df` DataFrame.
5. Add the `CoinName` column that holds the names of the cryptocurrencies, which you created in Step 7 of Deliverable 1, to the `clustered_df`.
6. Add another new column to the `clustered_df` named `Class` that holds the predictions, i.e., `model.labels_`, from Step 3.

Your `clustered_df` DataFrame should look like the image below:

	Algorithm	ProofType	TotalCoinsMined	TotalCoinSupply	PC 1	PC 2	PC 3	CoinName	Class
42	Scrypt	PoW/PoS	42	42	-0.335587	1.060721	-0.442783	42 Coin	0
404	Scrypt	PoW/PoS	1.00862e+09	532000000	-0.319468	1.060956	-0.443089	404Coin	0
1337	X13	PoW/PoS	2.92794e+10	314159265359	2.298643	1.616176	-0.543968	EliteCoin	0
BTC	SHA-256	PoW	17918662	21000000	-0.146783	-1.352007	0.158842	Bitcoin	1
ETH	Ethash	PoW	1.07626e+08	0	-0.134787	-1.929485	0.248370	Ethereum	1
LTC	Scrypt	PoW	6.30392e+07	84000000	-0.157018	-1.083072	0.028331	Litecoin	1
DASH	X11	PoW/PoS	9.02401e+06	22000000	-0.409440	1.226179	-0.562194	Dash	0
XMR	CryptoNight-V7	PoW	1.71937e+07	0	-0.165529	-2.462011	0.404730	Monero	1
ETC	Ethash	PoW	113255332	210000000	-0.133224	-1.929573	0.248345	Ethereum Classic	1
ZEC	Equihash	PoW	7.35252e+06	21000000	-0.128082	-1.987514	0.351649	ZCash	1

Save your `crypto_clustering.ipynb` file to your Cryptocurrencies folder.

Deliverable 3 Requirements

You will earn a perfect score for Deliverable 3 by completing all requirements below:

- The K-means algorithm is used to cluster the cryptocurrencies using the PCA data, where the following steps have been completed:
  - An elbow curve is created using `hvPlot` to find the best value for K **(10 pt)**
  - Predictions are made on the K clusters of the cryptocurrencies' data **(5 pt)**
  - A new DataFrame is created with the same index as the `crypto_df` DataFrame and has the following columns: `Algorithm`, `ProofType`, `TotalCoinsMined`, `TotalCoinSupply`, `PC 1`, `PC 2`, `PC 3`, `CoinName`, and `Class` **(5 pt)**

Deliverable 4: Visualizing Cryptocurrencies Results (30 points)

Deliverable 4 Instructions

Using your knowledge of creating scatter plots with Plotly Express and `hvplot`, you'll visualize the distinct groups that correspond to the three principal components you created in Deliverable 2, then you'll create a table with all the currently tradable cryptocurrencies using the `hvplot.table()` function.

REWIND

For this deliverable, you've already done the following in this module:

- **Lesson 18.3.2:** Create a scatter plot using `hvplot`
- **Lesson 18.4.2:** Create a 3D scatter plot with Plotly Express

Follow the instructions below and use the information in the `crypto_clustering_starter_code.ipynb` file to complete Deliverable 4.

1. Continue using the `crypto_clustering.ipynb` file from Deliverable 3 where you have predicted the K clusters for the cryptocurrencies' data.
2. Create a 3D scatter plot using the Plotly Express `scatter_3d()` function to plot the three clusters from the `clustered_df` DataFrame.
3. Add the `CoinName` and `Algorithm` columns to the `hover_name` and `hover_data` parameters, respectively, so each data point shows the CoinName and Algorithm on hover.

If you'd like a hint on how to add additional parameters to a Plotly Express 3D scatter plot, that's totally okay. If not, that's great too. You can always revisit this later if you change your mind.

SHOW HINT

4. Create a table with tradable cryptocurrencies using the `hvplot.table()` function.

If you'd like a hint on how to use the `hvplot.table()` function, that's totally okay. If not, that's great too. You can always revisit this later if you change your mind.

SHOW HINT

Your table should look like the table in the image below:

#	CoinName	Algorithm	ProofType	TotalCoinSupply	TotalCoinsMined	Class
0	42 Coin	Scrypt	PoW/PoS	42	41.999954	1
1	404Coin	Scrypt	PoW/PoS	532000000	1,055,184,902.04	1
2	EliteCoin	X13	PoW/PoS	314159265359	29,279,424,622.5027	1
3	Bitcoin	SHA-256	PoW	21000000	17,927,175.0	0
4	Ethereum	Ethash	PoW	0	107,684,222.6865	0
5	Litecoin	Scrypt	PoW	84000000	63,039,243.300005	0
6	Dash	X11	PoW/PoS	22000000	9,031,294.375634	1
7	Monero	CryptoNight-V7	PoW	0	17,201,143.144913	0
8	Ethereum Classic	Ethash	PoW	210000000	113,359,703.0	0
9	ZCash	Equihash	PoW	21000000	7,383,056.25	0
10	Bitshares	SHA-512	PoS	3600570502	2,741,570,000.0	1

5. Print the total number of tradable cryptocurrencies in the `clustered_df` DataFrame.
6. Use the `MinMaxScaler().fit_transform` method to scale the `TotalCoinSupply` and `TotalCoinsMined` columns between the given range of zero and one.

If you'd like a hint on how to use the `MinMaxScaler().fit_transform` method to scale the "TotalCoinSupply" and "TotalCoinsMined" columns, that's totally okay. If not, that's great too. You can always revisit this later if you change your mind.

SHOW HINT

7. Create a new DataFrame using the `clustered_df` DataFrame index that contains the scaled data you created in Step 5.
8. Add the `CoinName` column from the `clustered_df` DataFrame to the new DataFrame.
9. Add the `Class` column from the `clustered_df` DataFrame to the new DataFrame.

Your new DataFrame should look similar to the image below:

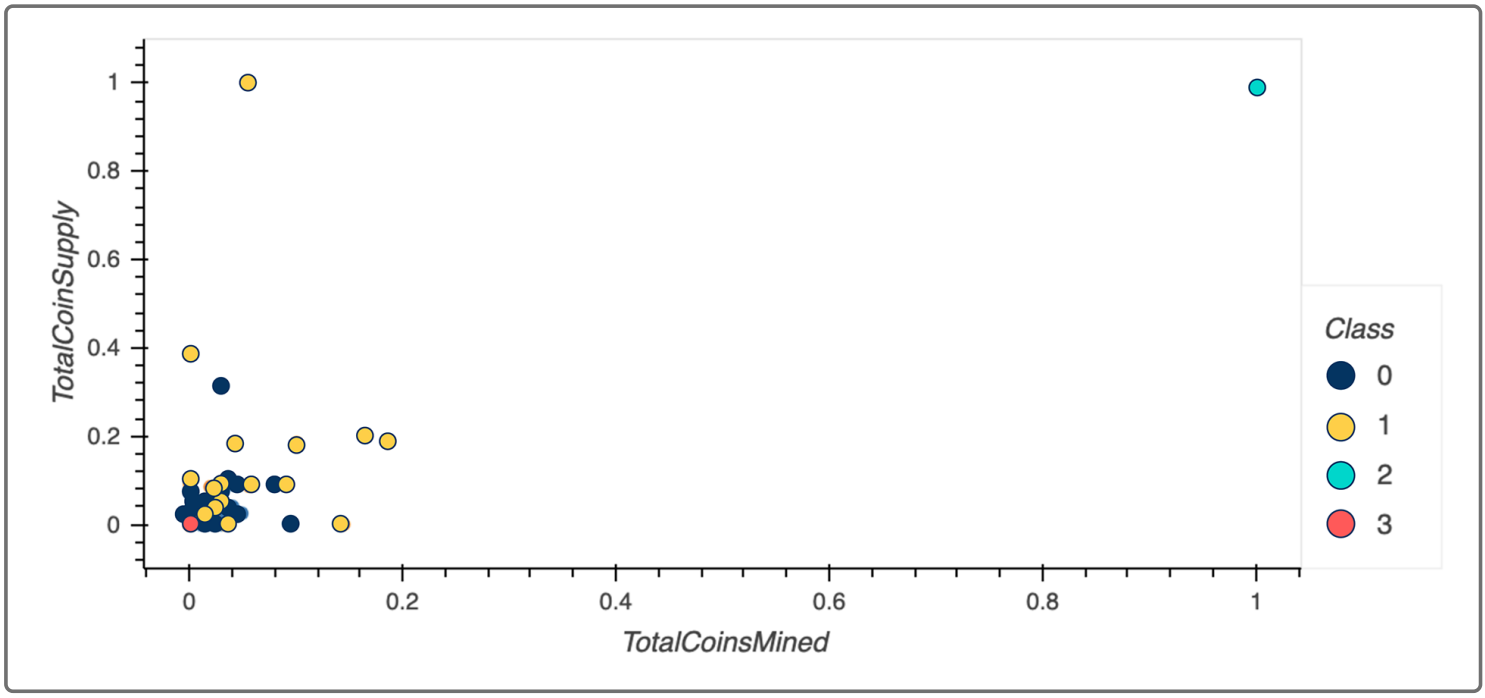
	TotalCoinSupply	TotalCoinsMined	CoinName	Class
42	4.200000e-11	0.000000	42 Coin	0
404	5.320000e-04	0.001066	404Coin	0
1337	3.141593e-01	0.029576	EliteCoin	0
BTC	2.100000e-05	0.000018	Bitcoin	1
ETH	0.000000e+00	0.000109	Ethereum	1
LTC	8.400000e-05	0.000064	Litecoin	1
DASH	2.200000e-05	0.000009	Dash	0
XMR	0.000000e+00	0.000017	Monero	1
ETC	2.100000e-04	0.000115	Ethereum Classic	1
ZEC	2.100000e-05	0.000007	ZCash	1

10. Create an `hvplot` scatter plot with x="TotalCoinsMined", y="TotalCoinSupply", and by="Class", and have it show the `CoinName` when you hover over the the data point.

If you'd like a hint on how to add the `CoinName` column data when you hover over a data point, that's totally okay. If not, that's great too. You can always revisit this later if you change your mind.

SHOW HINT

Your scatter plot should look similar to the image below:



Save your `crypto_clustering.ipynb` file to your Cryptocurrencies folder.

Deliverable 4 Requirements

You will earn a perfect score for Deliverable 4 by completing all requirements below:

- The clusters are plotted using a 3D scatter plot, and each data point shows the CoinName and Algorithm on hover **(10 pt)**
- A table with tradable cryptocurrencies is created using the `hvplot.table()` function **(3 pt)**
- The total number of tradable cryptocurrencies is printed **(2 pt)**
- A DataFrame is created that contains the `clustered_df` DataFrame index, the scaled data, and the `CoinName` and `Class` columns **(5 pt)**
- A `hvplot` scatter plot is created where the X-axis is "TotalCoinsMined", the Y-axis is "TotalCoinSupply", the data is ordered by "Class", and it shows the CoinName when you hover over the data point **(10 pt)**

Submission

Once you're ready to submit, make sure to check your work against the rubric to ensure you are meeting the requirements for this Challenge one final time. It's easy to overlook items when you're in the zone!

As a reminder, the deliverables for this Challenge are as follows:

- Deliverable 1: Preprocessing the Data for PCA
- Deliverable 2: Reducing Data Dimensions Using PCA
- Deliverable 3: Clustering Cryptocurrencies Using K-means
- Deliverable 4: Visualizing Cryptocurrencies Results

Upload the following to your Cryptocurrencies GitHub repository:

- Your `crypto_clustering.ipynb` file.
- A README.md that includes the purpose of the repository and short description of what was accomplished. Although there is no graded written analysis for this challenge, it is encouraged and good practice to add a brief description of your project.

To submit your challenge assignment for grading in Bootcamp Spot, click Start Assignment, click the Website URL tab, then provide the URL of your Cryptocurrencies GitHub repository, and then click Submit. Comments are disabled for graded submissions in BootCampSpot. If you have questions about your feedback, please notify your instructional staff or the Student Success Manager. If you would like to resubmit your work for an improved grade, you can use the **Re-Submit Assignment** button to upload new links. You may resubmit up to 3 times for a total of 4 submissions.

IMPORTANT

Once you receive feedback on your Challenge, make any suggested updates or adjustments to your work. Then, add this week's Challenge to your professional portfolio.

NOTE

You are allowed to miss up to two Challenge assignments and still earn your certificate. If you complete all Challenge assignments, your lowest two grades will be dropped. If you wish to skip this assignment, click Next, and move on to the next Module.

Module-18 Rubric

Criteria	Ratings					Pts
Deliverable 1: Preprocessing the Data for PCA	**30 to >27.0 pts Demonstrating Proficiency** The following preprocessing steps have been performed on the crypto_df DataFrame: (15 pt) ✓ All cryptocurrencies that are not being traded are removed ✓ The IsTrading column is dropped ✓ All cryptocurrencies with at least one null value are dropped ✓ All the rows that do not have coins being mined are removed ✓ The CoinName column is dropped from the crypto_df DataFrame AND all the following have been completed. ✓ A new DataFrame is created that stores the names of all cryptocurrencies from the CoinName column and has the index from the crypto_df DataFrame (5 pt) ✓ The get_dummies() method is used to create variables for all of the text features, which are then stored in a new DataFrame, X (5 pt) ✓ The features from the X DataFrame have been	**27 to >24.0 pts Approaching Proficiency** The following preprocessing steps have been performed on the crypto_df DataFrame: (15 pt) ✓ All cryptocurrencies that are not being traded are removed ✓ The IsTrading column is dropped ✓ All cryptocurrencies with at least one null value are dropped ✓ All the rows that do not have coins being mined are removed ✓ The CoinName column is dropped from the crypto_df DataFrame AND all the following have been completed. ✓ A new DataFrame is created that stores the names of all cryptocurrencies from the CoinName column and has the index from the crypto_df DataFrame (5 pt) ✓ The get_dummies() method is used to create variables for the TWO text features, which are then stored in a new DataFrame, X (5 pt) ✓ The features from the X DataFrame have been	**24 to >20.0 pts Developing Proficiency** The following preprocessing steps have been performed on the crypto_df DataFrame: (13 pt) ✓ All cryptocurrencies that are not being traded are removed ✓ The IsTrading column is dropped ✓ All cryptocurrencies with at least one null value are dropped ✓ All the rows that do not have coins being mined are removed AND all the following have been completed. ✓ A new DataFrame is created that stores the names of all cryptocurrencies with the CoinName column and has the index from the crypto_df DataFrame (5 pt) ✓ The get_dummies() method is used to create variables for the ONE of the TWO text features, which are then stored in a new DataFrame, X (3 pt) ✓ The features from the X DataFrame have been	**20 to >0.0 pts Emerging** The following preprocessing steps have been performed on the crypto_df DataFrame: (12 pt) ✓ All cryptocurrencies that are not being traded are removed ✓ All cryptocurrencies with at least one null value are dropped ✓ All the rows that do not have coins being mined are removed AND all the following have been completed. ✓ A new DataFrame is created that stores the names of all cryptocurrencies with the CoinName column BUT does not have the index from the crypto_df DataFrame (2 pt) ✓ The get_dummies() method is used to create variables for the ONE of the TWO text features, which are then stored in a new DataFrame, X (3 pt) ✓ The features from the X DataFrame have been	**0 pts Incomplete**	30 pts
Deliverable 2: Reducing Data Dimensions Using PCA	**20 to >18.0 pts Demonstrating Proficiency** ✓ The PCA algorithm reduces the dimensions of the X DataFrame down to three principal components. (10 pt) ✓ The pcs_df DataFrame is created (5 pt) And, all the following are completed: ✓ The pcs_df DataFrame has three columns, PC 1, PC 2, and PC 3 (2 pt) ✓ The	**18 to >16.0 pts Approaching Proficiency** ✓ The PCA algorithm reduces the dimensions of the X DataFrame down to three principal components. (10 pt) ✓ The pcs_df DataFrame is created (5 pt) And, ONE of the following are completed: ✓ The pcs_df DataFrame has three columns, PC 1, PC 2, and PC 3 (2 pt) ✓ The	**16 to >14.0 pts Developing Proficiency** ✓ The PCA algorithm reduces the dimensions of the X DataFrame down to three principal components. (10 pt) ✓ The pcs_df DataFrame is created (5 pt) ✓ Code is written to add the three features from the X DataFrame, PC 1, PC 2, and PC 3 have been	**14 to >0.0 pts Emerging** ✓ The PCA algorithm reduces the dimensions of the X DataFrame down to three principal components. (10 pt) ✓ Code is written to create the pcs_df DataFrame with the three columns, PC 1, been	**0 pts Incomplete**	20 pts

Criteria	Ratings				Pts
Deliverable 3: Clustering Cryptocurrencies Using K-means	**20 to >18.0 pts** **Demonstrating Proficiency** ✓ Data is read into a DataFrame (5 pt) ✓ An Elbow Curve is created using hvPlot to find the best value for K (10 pt) ✓ Predictions are made on the K clusters of the cryptocurrencies' data (5 pt) ✓ A new DataFrame is created with the same index as the crypto_df DataFrame and has NINE columns (5 pt)	**18 to >15.0 pts** **Approaching Proficiency** ✓ Data is read into a DataFrame (5 pt) ✓ An Elbow Curve is created using hvPlot to find the best value for K (10 pt) ✓ Predictions are made on the K clusters of the cryptocurrencies' data (5 pt) ✓ A new DataFrame is created with the same index as the crypto_df DataFrame and has SEVEN of the NINE columns (3 pt)	**15 to >12.0 pts** **Developing Proficiency** ✓ Data is read into a DataFrame (1 pt) ✓ An Elbow Curve is created using hvPlot to find the best value for K (10 pt) ✓ Predictions are made on the K clusters of the cryptocurrencies' data (5 pt) ✓ A new DataFrame is created with the same index as the crypto_df DataFrame and has SEVEN of the NINE columns (3 pt)	**12 to >0.0 pts** **Emerging** ✓ Data is read into a DataFrame (1 pt) ✓ An Elbow Curve is created using hvPlot to find the best value for K (10 pt) ✓ Code is written to make the predictions on the K clusters of the cryptocurrencies' data (2 pt) ✓ The clusters are plotted using a 3D-Scatter and each data point shows the CoinName and Algorithm on hover (10 pt) ✓ A table with tradable cryptocurrencies is created using the hvplot.table() function (3 pt) ✓ The total number of tradable cryptocurrencies is printed (2 pt) ✓ A DataFrame is created that contains the scaled data and the clustered_df DataFrame index. (2 pt) ✓ A hvplot scatter plot is created where the X-axis is "TotalCoinsMined", the Y-axis is "TotalCoinSupply" (5 pt)	**0 pts** **Incomplete**	20 pts
Deliverable 4: Visualizing Cryptocurrencies Results	**30 to >27.0 pts** **Demonstrating Proficiency** ✓ The clusters are plotted using a 3D-Scatter and each data point shows the CoinName and Algorithm on hover (10 pt) ✓ A table with tradable cryptocurrencies is created using the hvplot.table() function (3 pt) ✓ The total number of tradable cryptocurrencies is printed (2 pt) ✓ A DataFrame is created that contains the clustered_df DataFrame index, the scaled data, and the "CoinName" and "Class" columns (5 pt) ✓ A hvplot scatter plot is created where the X-axis is "TotalCoinsMined", the Y-axis is "TotalCoinSupply", and the data is ordered by	**27 to >25.0 pts** **Approaching Proficiency** ✓ The clusters are plotted using a 3D-Scatter and each data point shows the CoinName and Algorithm on hover (10 pt) ✓ A table with tradable cryptocurrencies is created using the hvplot.table() function (3 pt) ✓ The total number of tradable cryptocurrencies is printed (2 pt) ✓ A DataFrame is created that contains the clustered_df DataFrame index, the scaled data, and the "CoinName" and "Class" columns (5 pt) ✓ A hvplot scatter plot is created where the X-axis is "TotalCoinsMined", the Y-axis is "TotalCoinSupply, and the data is ordered by	**25 to >22.0 pts** **Developing Proficiency** ✓ The clusters are plotted using a 3D-Scatter and each data point shows the CoinName and Algorithm on hover (10 pt) ✓ A table with tradable cryptocurrencies is created using the hvplot.table() function (3 pt) ✓ The total number of tradable cryptocurrencies is printed (2 pt) ✓ A DataFrame is created that contains the scaled data, the clustered_df DataFrame index, and the "CoinName" column (3 pt) ✓ A hvplot scatter plot is created where the X-axis is "TotalCoinsMined", the Y-axis is "TotalCoinSupply, and when you hover over the the	**22 to >0.0 pts** **Emerging** ✓ The clusters are plotted using a 3D-Scatter and each data point shows the CoinName and Algorithm on hover (10 pt) ✓ A table with tradable cryptocurrencies is created using the hvplot.table() function (3 pt) ✓ The total number of tradable cryptocurrencies is printed (2 pt) ✓ A DataFrame is created that contains the scaled data and the clustered_df DataFrame index. (2 pt) ✓ A hvplot scatter plot is created where the X-axis is "TotalCoinsMined", the Y-axis is "TotalCoinSupply" (5 pt)	**0 pts** **Incomplete**	30 pts
					Total Points: 100

“Class”, and when you hover over the the data it shows the “CoinName” (10 pt)	“Class”, but the “CoinName” doesn’t appear on hover (7 pt)	data it shows the “CoinName” (7 pt)