Capstone Project

Machine Learning Engineer Nanodegree

Vaishali Sutaria
March 12, 2018

Definition

Project Overview

This project is based on the concept of project delivery time allocated for New Site Build sites in the Telecommunication industry and the various major milestones in between that contribute to the success or failure of delivering on time. The idea is this project can be applied to any type of project with milestones.

This proposal is specifically based on cycle time (in days) it takes for new construction site in a Telecommunication industry to build. Building a New Site in different cities have different jurisdiction requirements and it takes different amount of time to complete. Some jurisdictions are more difficult then others which is one of the major milestones in the construction process contributing to the delivery time. There are other milestones like candidate approval, zoning submitted, zoning approval, construction start and construction complete which also contribute to the delivery date.

When companies allocate funding for any given project it comes with expectations to complete in 2 years (730 days) called the delivery date. The company generally spends hundreds of dollars investing and (reserving)forecasting funds on a new site build project based on the delivery date and year. However, some of the major milestones take lot longer in cycle times and sometimes unpredictable which delays the build.

This is troublesome because the current approach is like an open check book and open risk. In order to reduce the uncertainties of the delivery date and spending additional funding and resources I have created this proposal. I would like to propose that the Telecommunication Companies invest ahead of time for difficult jurisdictions or increase the delivery dates. To get approval for such recommendation to be made we need to quantify and look at historical data on the performances in the different jurisdictions and classify them. This will allow the company to direct the resources accordingly. This will also help improve the forecast accuracy for finance team on the New Site Build Projects.

In this project I would like to analyze a fictitious dataset where I will analyze the number of days taken to complete 9 major milestones(features).

The benchmark standard or goal for the given state is to finish the build in 2 years, I am using 730 days for this project. However, knowing that not everything gets build in 2 years logically we should be providing funding ahead of time for those jurisdictions. If I make such statement it will not be enough to convince the leadership team so I have created a historic fact based Machine Learning Engineering model to demonstrate the same based on data and algorithms that suggest the same.

In this project I have applied what I have learned from the Udacity program and identified the important features based on relevance. I create clusters of the types of jurisdictions based on the dataset. The end goal is to identify which cities / jurisdictions we should either increase/adjust the delivery date or we should provide funding in advance.

Here are the 10 major milestones (features) I have in my Dataset:

Days to Vendor Handoff	MS1
Zoning Submitted	MS2
Zoning Approved	MS3
Candidate Identified	MS4
Candidate Approved	MS5
Site Acq /Jurisdiction Approved	MS6
CX Start	MS7
Tower Ready	MS8
CX Complete	MS9
ON AIR	MS10

Here are the steps I have taken to achieve this:
Load Data
Explore Data
Prepare Data
Normalizing the data
Naïve Predictor Performance
PCA
Dimensionality Reduction Technique
Log Transform
Clustering
Final Model Evaluation

Links to Similar projects:

1. Similar example of using k-means clustering for customer segmentation.

http://www.kimberlycoffey.com/blog/2016/8/k-means-clustering-for-customer-segmentation

2. Another Customer Segmentation project on Banking Customers.

https://www.linkedin.com/pulse/banking-customer-segmentation-machine-learning-sachin-jahagirdar

3. Finding your best customers, slightly different approach RFM Score Calculation.

https://towardsdatascience.com/find-your-best-customers-with-customer-segmentation-in-python-61d602f9eee6

Problem Statement

The problem is building a new site is taking lot longer than expected. Ideally, 2 years is the benchmark to build new sites for a Telecommunication industry. However, the reality is not all projects released to build in 2 years are build in time. There are sites that will build before time and there are sites that will build after the delivery date. The delay and acceleration of the project delivery has a financial impact. The fundamental way of funding strategy needs to be changed by jurisdiction not by cycle times. I want to identify by segmenting the customers which cluster should be given funding to begin the pre funding jurisdiction process. This approach will help ensure the New site is delivered on or before time.

Metrics

I have used Silhouette Score as my evaluation metric.

When the ground truth labels are not known, evaluation needs to be performed using the model itself. Silhouette Coefficient is an example of such evaluation. A higher Silhouette Coefficient score relates to a model with better defined clusters. The score Silhouette score is defined for each sample and is composed of 2 scores:

- The mean distance between a sample and all other points in the same class.
- The mean distance between a sample and all the other points in the next nearest cluster.

  S for a single sample is then given as:

$$s = \frac{b - a}{max(a, b)}$$

I selected the Silhouette Coefficient because of its ability to quantify the goodness of clustering by calculating each data points. The Silhouette Coefficient for a data point measures how similar it is to its assigned cluster. -1 (not similar) is assigned to incorrect clustering and 1 (similar) is assigned for highly dense clustering. Scores around 0 indicates overlapping clusters. Therefore, providing a mean silhouette coefficient provides simple scoring method of a given clustering and also helps identify the various segments hidden in the data. This is very beneficial because this project is about clustering and scoring the accuracy of the clustering is the metric I was looking for. I found it to be perfect fit for my model.

In my code cell I have fit the reduced data set and assigned them to predict a cluster. Second found the centers of the clusters using the algorithm's attribute. Third predicted the clusters for each sample data points. Lastly I calculated the Silhouette score of the reduced data.

Here are my results: Silhouette Scores for different clusters. You will see that 2 clusters have the best Silhouette Score of 43.25%. I ended up choosing 3 clusters for this project which has a Silhouette Score of 33.67%

```
# 2 = 0.43258002255113626
# 3 = 0.33666175827871947
# 4 = 0.35746014747047655
# 6 = 0.36285833356406383
# 9 = 0.36788137744626376
# 10 = 0.3807522100340986
```

Analysis

Data Exploration

I have used a randomly generated dataset of 579 samples with 10 milestones as features. In general I am very familiar with the various milestones involved with building a New Site so it was easy for me to choose the top 10 milestones to analyze. I then pulled all the cities in California and generated records for 260+ cities and assigned various cycle times based on every possible scenario that I am familiar with. My role is in Planning which gives me the advantage to see the life cycle of a New Site project from Funding to ON AIR. I used this knowledge to develop some scenarios for the data set. One example would be a situation where the candidate fails and now the project takes longer to find another candidate. This instance would have longer then expected cycle time so I created records for such scenario.

Sometimes the situations or milestones are skewed because of database issues and sometimes the milestones are skewed because of the delay.

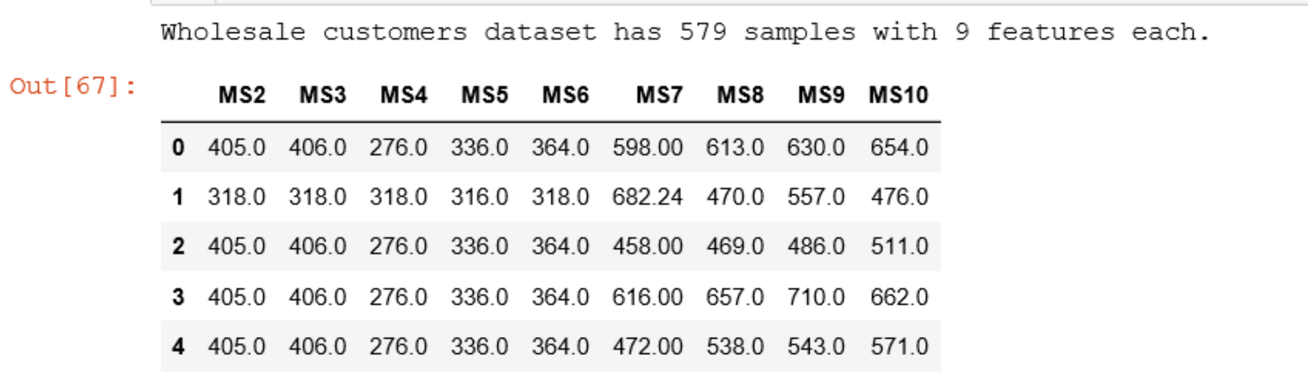
Descriptive Stats:
MS10 - ON AIR is the most important milestone as this is the actual delivery date. Below I have provided average cycle times for the last 5 milestones for the first 5 cities/jurisdictions. You can quickly spot out that Alpine has higher average of 1004 days for ON AIR vs other cities. Similarly, you will see Aliso Viejo, average days for ON AIR is 646. This might be a candidate for delivery date reduction.

City	Average of MS6	Average of MS7	Average of MS8	Average of MS9	Average of MS10
ALAMEDA	551	561	688	760	734
ALBANY	649	661	800	943	845
ALISO VIEJO	438	500	623	686	646
ALPINE	772	882	965	985	1004
ANAHEIM	508	571	680	779	704

Key – Key is a record created to represent a New Site Build
City – City
County – County
State – State of build
Zip – Zip code

Days to Vendor Handoff	MS1
Zoning Submitted	MS2
Zoning Approved	MS3
Candidate Identified	MS4
Candidate Approved	MS5
Site Acq /Jurisdiction Approved	MS6
CX Start	MS7
Tower Ready	MS8
CX Complete	MS9
ON AIR	MS10

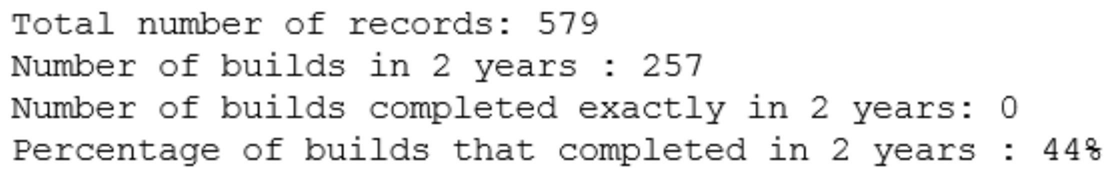
Here is a data sample:



```
Wholesale customers dataset has 579 samples with 9 features each.
```

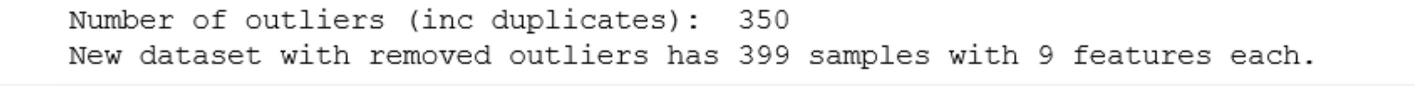
Out[67]:

	MS2	MS3	MS4	MS5	MS6	MS7	MS8	MS9	MS10
0	405.0	406.0	276.0	336.0	364.0	598.00	613.0	630.0	654.0
1	318.0	318.0	318.0	316.0	318.0	682.24	470.0	557.0	476.0
2	405.0	406.0	276.0	336.0	364.0	458.00	469.0	486.0	511.0
3	405.0	406.0	276.0	336.0	364.0	616.00	657.0	710.0	662.0
4	405.0	406.0	276.0	336.0	364.0	472.00	538.0	543.0	571.0



```
Total number of records: 579
Number of builds in 2 years : 257
Number of builds completed exactly in 2 years: 0
Percentage of builds that completed in 2 years : 44%
```

I removed the outliers detected:



```
Number of outliers (inc duplicates):  350
New dataset with removed outliers has 399 samples with 9 features each.
```

For the purposes of this exercise I dropped Key – as this is just a label, city, county, state, zip code and MS1 milestone. As I only wanted to focus on important features. MS1 is the vendor handoff point which can be biased and related to administrative delays and not true delays based on requirements, jurisdictions etc. so I dropped this feature for this analysis.

Input file Clean_Data_California.csv was used as one of the input.
Visuals.py is a supplementary file to help with the visuals.

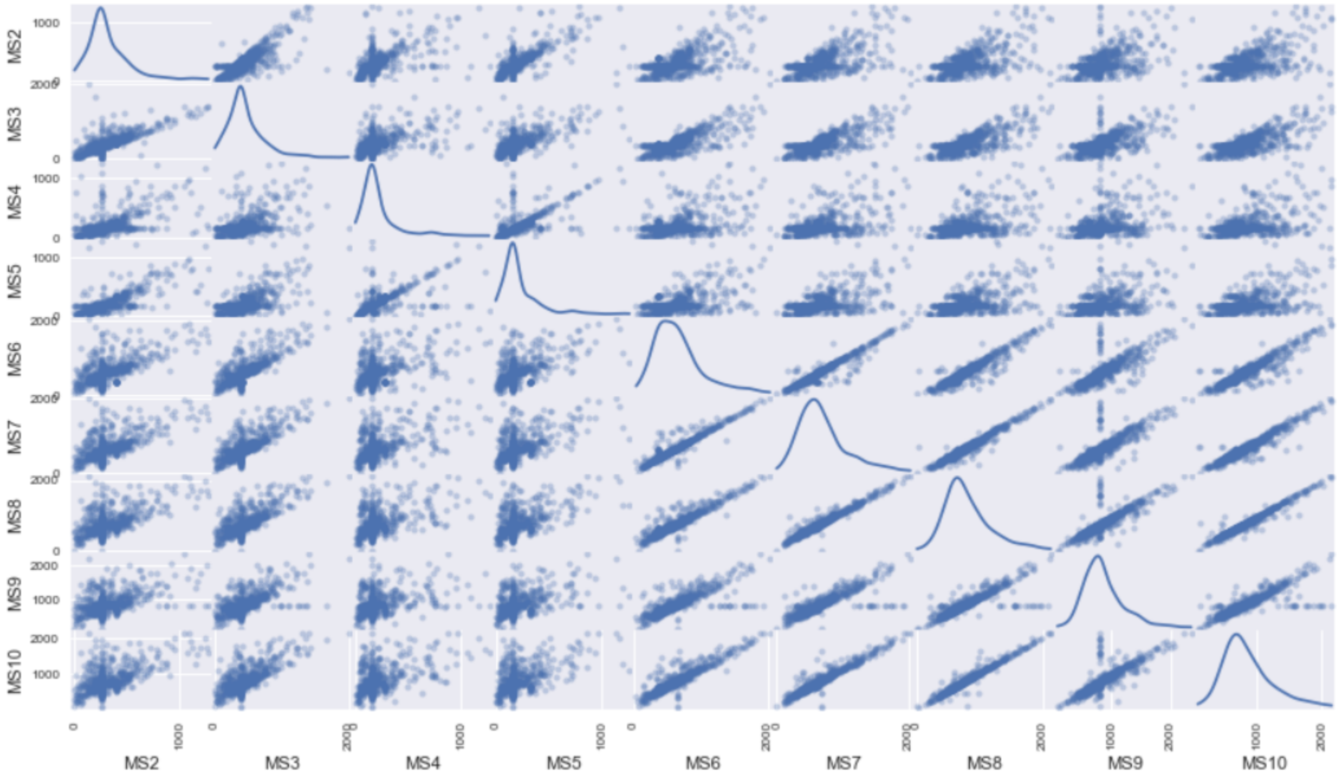
Based on my knowledge of this industry I know that MS6 Site Acq / Jurisdiction approval are the big contributing factors to the delay. But what I learned during this process that MS6 is very highly correlated with MS7 (construction start) and MS8 (Tower Ready).

MS6 (Site Acq/Jurisdiction approval, MS7 (Construction Start) and MS8 (Tower Ready) seem to show a greater degree of correlation. So, anyone of these features could likely be dropped because they are not unique or valuable in helping me establish the profile of the data. This is just an observation note.

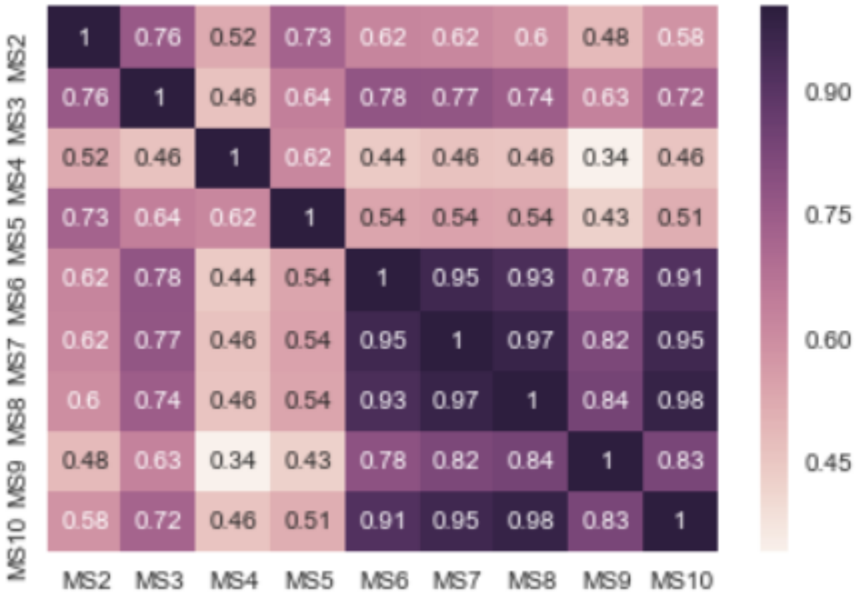
I also attempted to predict the MS6 milestone which is the time it takes to complete the site acquisition process. The score is 83.46%. Since the score is high and closer to 1 tells me that other features correlate closely with the site acquisition process MS6 milestone. Likely this milestone will not be providing additional insights. Although it is ok to drop this feature i want to keep it because I want it to be included in the dimensioning step.

Exploratory Visualization

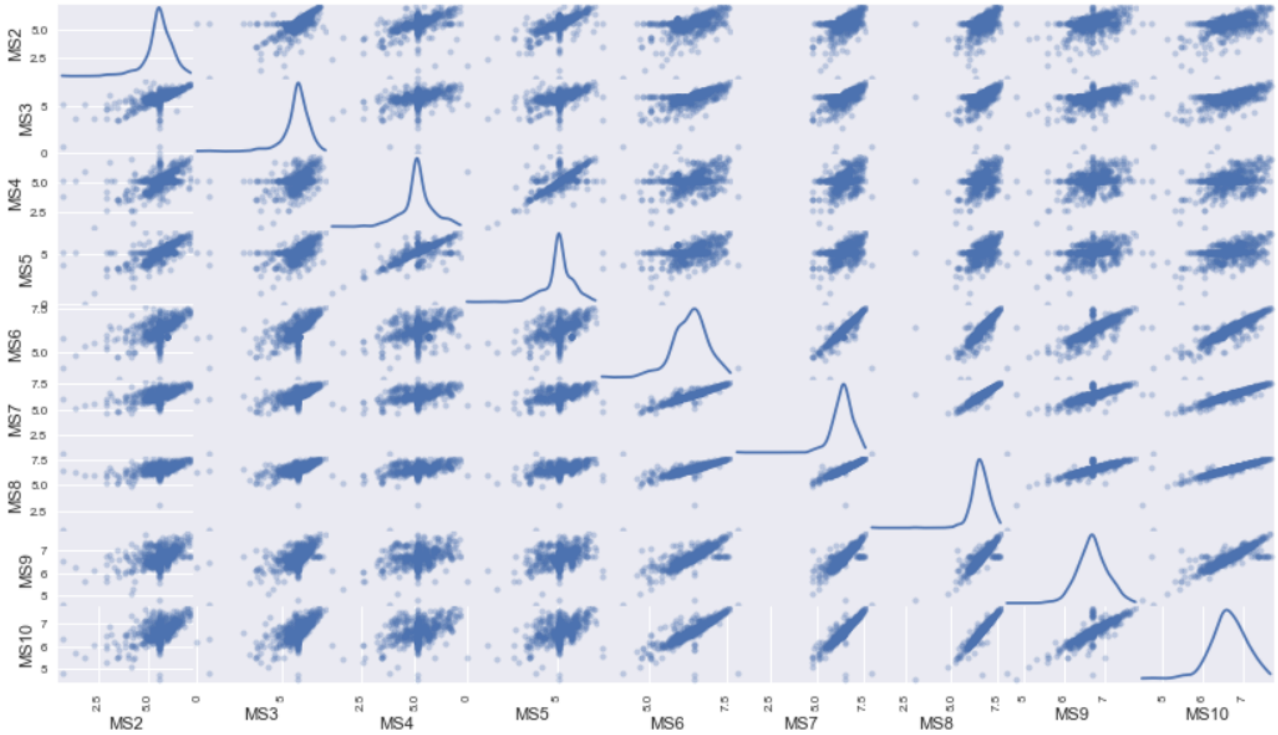
I used Scatter Matrix for Data Visualization:



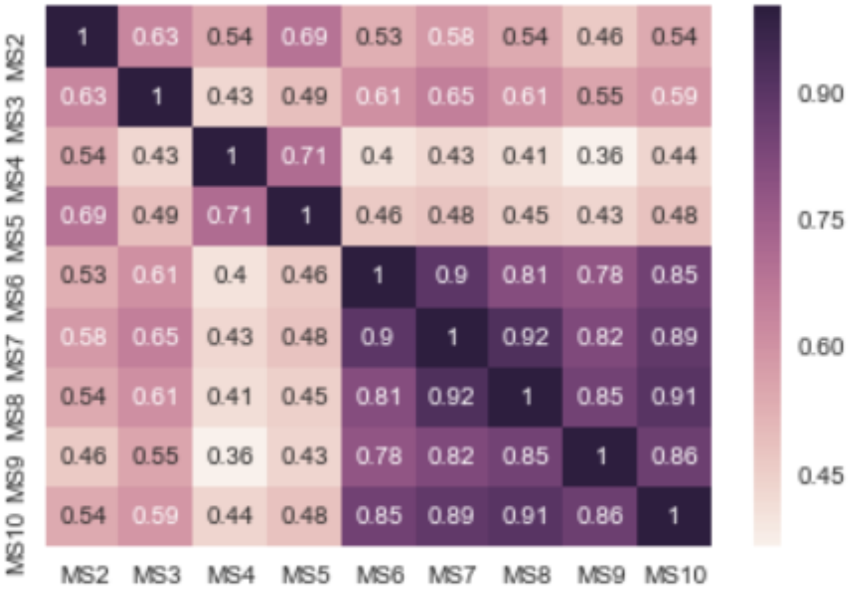
To get a better understanding of the dataset, we can construct a scatter matrix of each of the 9 features present in the data to see the correlation between features.



I also used a heatmap because it gives me a clear picture of correlation.



Here I am reviewing the scatter matrix for the newly transformed feature.



The same with heatmap created for the newly transformed feature. You can see change in the correlation.

The key takeaway here is to see the transformation of the features after natural logarithm scaling. If you look at the first scatter matrix you will see the blue dots more scattered spread out, but the second scatter matrix will see the data points are more normalized and grouped.

Similarly, in the heat maps you will see in the first chart that MS6 is highly correlated with MS7, MS8 and MS10 by 95%, 93% and 91% consecutively. The second heat map after the feature transformation still shows high correlation but slightly different MS6 to MS7,MS8 and MS10 now show 90%, 81% and 85%.

Looking at the charts you will see that the data has quite a bit of variance so we need an algorithm that will identify the principal components of the data. The main idea of Principal Component Analysis (PCA) is to reduce the dimensionality of the dataset consisting of many variables correlated with each other. Either the correlation is heavy or light, yet it retains the variation present in the dataset. This analysis helped me decide choosing PCA.

Algorithms and Techniques

- I used Feature transformation PCA technique in my project because my data is correlated. Meaning my features are not mutually independent. The problem that I am trying to solve is to propose a strategy for funding. To get to this stage, I need to create clusters of my data and this can be done only when we can get to dimensions that explain the variance very well. PCA technique allows us to see the various dimensions of the dataset with % of variance explained. This technique allowed me to narrow down and choose 1st and 2nd component that explained 80% of the variance explained below.

```
from sklearn.decomposition import PCA


# TODO: Apply PCA by fitting the good data with the same number of dimensions as features
pca = PCA().fit(good_data.values)

# TODO: Transform log_samples using the PCA fit above
pca_samples = pca.transform(log_samples)

# Generate PCA results plot
pca_results = vs.pca_results(good_data, pca)
```

If i look at the 1st and 2nd component 59.99% + 20.41% = 80.40% of the variance is explained. If we look at the first 4 components 59.99% + 20.41%+ 8.50% +4.10% = 93.00% of the variance is explained.

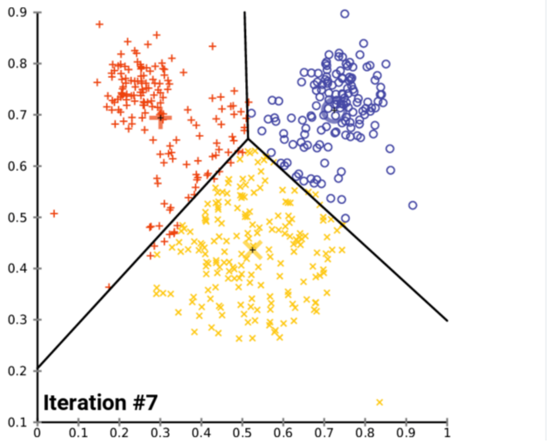
- Second I did a dimensionality reduction technique. Because a good amount of my variance in the data was explained by the 1st and 2nd component I decided to apply PCA by fitting good data with only 2 dimensions.

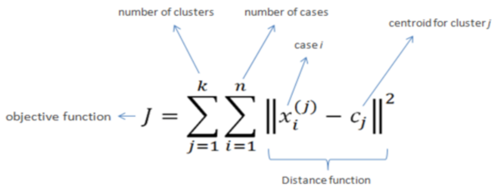
```
# TODO: Apply PCA by fitting the good data with only two dimensions
pca = PCA(n_components=2)
pca.fit(good_data)

# TODO: Transform the good data using the PCA fit above
reduced data = pca.transform(good_data)
```

```
# TODO: Transform log_samples using the PCA fit above
pca_samples = pca.transform(log_samples)

# Create a DataFrame for the reduced data
reduced data = pd.DataFrame(reduced data, columns = ['Dimension 1', 'Dimension 2'])
```

- In have also used K-Means clustering algorithm to identify the various segments hidden in the data. Then I recovered data points from the clusters to understand their significance by transforming them back into the original dimensions and scale. I used K-Means because it is easy to implement with large number of variables.
- K-means clustering is a type of unsupervised learning, which is used when you have unlabeled data. Meaning no categories or groups assigned to the data. The goal of this algorithm is to find groups in the data, with the number of groups represented by the variable K. the algorithm works iteratively to assign each data point to one of the K groups based on the features that are provided. Data points are clustered based on feature similarities. Generally, in the result of K-mean clustering algorithm the centroids of the k clusters can be used to label new data and Labels for the training data /each data point is assigned to a single cluster. So rather than defining groups before looking at the data clustering allows you to find and analyze the groups that have been formed.

Visual of K-means Clustering





Benchmark

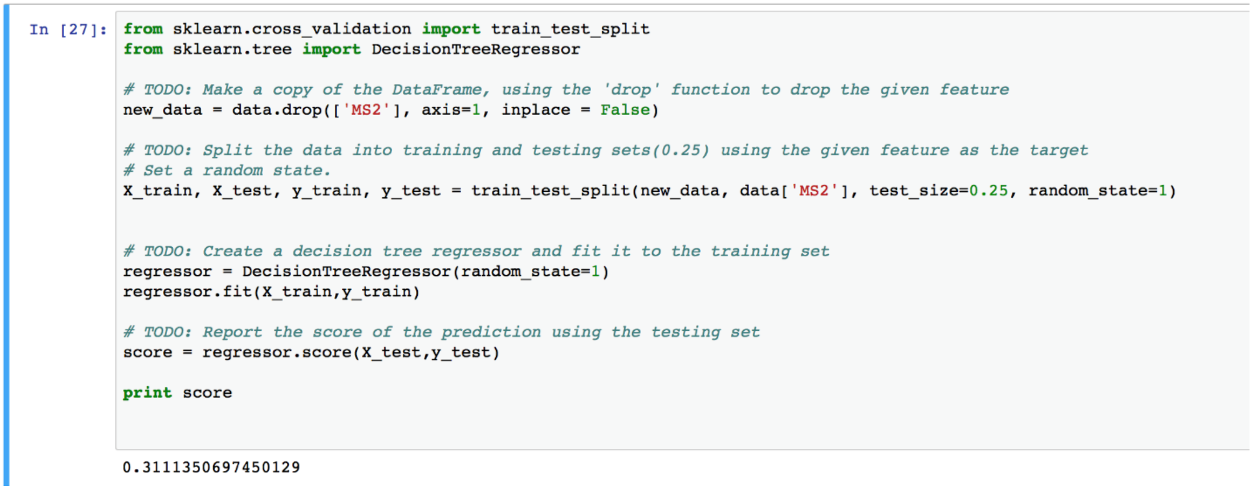
As an industry standard baseline benchmark, I have used 730 days/ 2 years to complete a new site build project. This is my first benchmark I can test algorithm performance against.

Second, I have used Naïve Predictor model as my base model for this project on my just one feature "MS10" from raw data. I simply took my data set and ran a basic test on it without preprocessing and removing outliers.

Methodology

Data Preprocessing

In my preprocessing step I reviewed the different milestones/features using the decision tree regressor technique by choosing one feature at a time to see the relevance. For Example, in my code I have run the decision tree regressor on MS6 Milestone (Site Acq/Jurisdiction Approval) the score was 83.78%. Since this score was high closer to 1 it tells me that other features in the dataset correlate closely with MS6. If I remove this Milestone I would likely still get similar clustering results. Similarly, if I run the decision tree regressor on MS2 I get a score of 31.11% which is quite low. The score is telling me that I should keep this milestone as it has low correlation with other milestones and provides insight about the dataset.
I looked at the feature distribution to see the different types of jurisdictions based on the data. Then I normalized the data by applying a logarithmic scaling both to the data and samples. Then I identified the outliers and removed them all. In total I removed 350 outliers.



```
In [27]: from sklearn.cross_validation import train_test_split
         from sklearn.tree import DecisionTreeRegressor

         # TODO: Make a copy of the DataFrame, using the 'drop' function to drop the given feature
         new_data = data.drop(['MS2'], axis=1, inplace = False)

         # TODO: Split the data into training and testing sets(0.25) using the given feature as the target
         # Set a random state.
         X_train, X_test, y_train, y_test = train_test_split(new_data, data['MS2'], test_size=0.25, random_state=1)


         # TODO: Create a decision tree regressor and fit it to the training set
         regressor = DecisionTreeRegressor(random_state=1)
         regressor.fit(X_train,y_train)

         # TODO: Report the score of the prediction using the testing set
         score = regressor.score(X_test,y_test)

         print score

         0.3111350697450129
```

I learned in the customer segment project that data preprocessing step of any analysis includes detecting and removing outliers. I did the same here where the outlier step is calculated as 1.5 times the interquartile range (IQR). A data point with a feature that is beyond an outlier step outside of IQR for that feature is considered abnormal.

I assigned 25th percentile, 75th percentile and assigned calculation of an outlier step and removed the outliers from the dataset.

Here we determine the min and max cutoffs for detecting the outliers. Q1 = 25% and Q3 = 75%
Step 1, get the Interquartile Range

$$IQR = Q_3 - Q_1$$

Step 2, calculate the upper and lower values

$$min = Q_1 - (IRQ \times 1.5)$$
$$max = Q_3 + (IQR \times 1.5)$$

We removed 350 outliers in this preprocessing step.

Implementation

1. I started this process by loading the dataset and and analyzing the data. I pulled some basic information about the data such as how many records and features. Then i looked at the number of builds in 2 years out of the 579 records. Only 44% of the builds were in 2 years. This means that 56% of the New Sites were funded with short delivery date.
2. Second I selected some random samples which I could check the clustering assignment after the training.
3. Then I created a Base Model Naïve Predictor to see if I was to train my dataset without any transformation how would it perform.
4. Then I checked the feature relevance of the milestones to understand if the data was correlated to other milestones or provide a different insight.
5. I used scatter and heat map to then review the data was spread and correlated to the other milestones.
6. Next I did data processing by applying natural logarithm to transform the data and soften the correlation.
7. At this point I break and check how the log transformation looks on the sample data.
8. Then I identify outlier step where extreme high and low values for each feature is removed using IQR step.
9. Then I do a quick review of the data after the outliers removed to see the mean, standard, min etc.…

10. Next I work on Feature Transformation by using PCA. In this section i have used PCA, principal component analysis to draw conclusions about the underlying structure. Since using the PCA on a dataset calculates the dimensions which best maximize variance, this will help find out which compound combinations of the features best describe the

jurisdictions. This section helped me understand which dimensions about the data best maximize the variance of features involved.

11 Next, I take the first few dimensions that add up to a good % of variance and do dimensionality reduction. In this project I used 2 dimensions.
12 Then I applied K-Means clustering and calculated the best Silhouette Score. 2 clusters give me the best Silhouette score but I decided to go with 3 clusters because I found the 3 clusters more insightful and created opportunity to discuss the sites that came ON AIR before 2 years. Those jurisdictions would allow for a savings opportunity by lower the delivery date.

13 Then I created visualizing clusters on the sample data to see if the model predicted accurately. Meaning show the sample in the right cluster.
14 Next I applied data recovery step and did an inverse transformation. I learned this feature in the Machine Learning program. Each cluster in the visualization has a central point. These are centers/means but not specifically data points rather they are averages of all the data points in the clusters. Since the data is reduced in dimension and scaled by a logarithm, we can recover the representative jurisdiction from these data points by applying the inverse transformations. Here I display the true centers. At this point I get the 3 clusters.

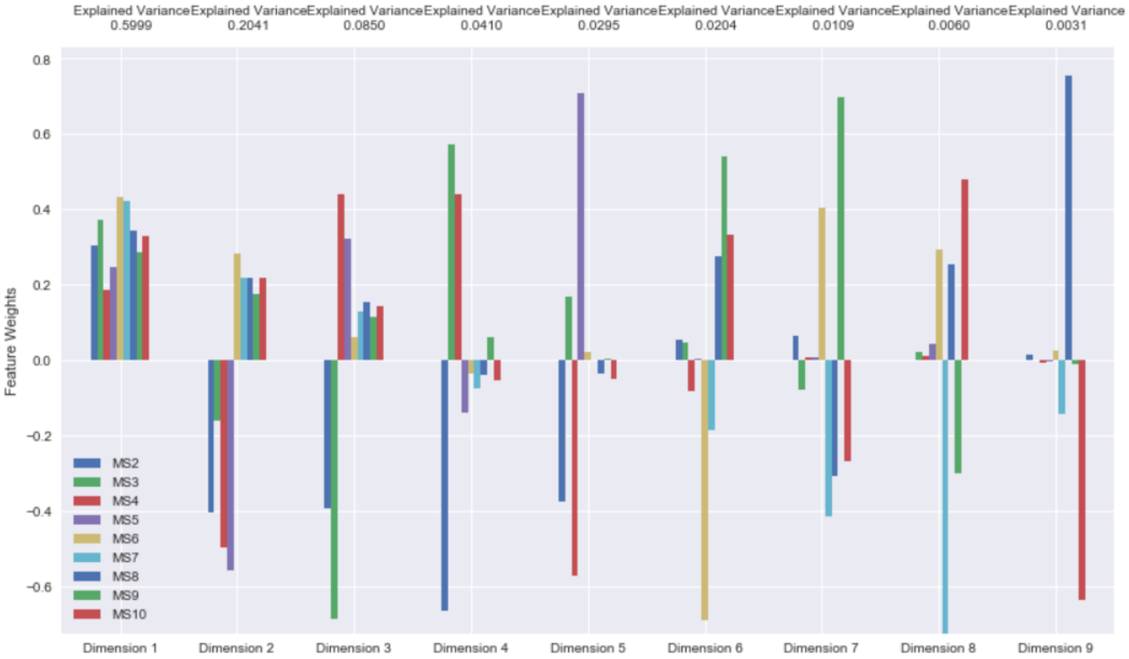
Segment 0 cluster tells me that the site acq/jurisdiction approval and on air time is well below 730 days/2 years. Perhaps these are candidates for consideration in reducing my delivery time.

Segment 1 cluster represent jurisdiction that will likely need to be considered for prefunding or increase in delivery time. The MS6 (Jurisdictional Approval) and MS10 (ON AIR) is way above the 730 days/2-year mark.

Segment 2 cluster tells me that we need not spend any time analyzing the strategy of funding on these. These can be funded as BAU efforts as they will be build in 2 years /730 days or close to it.

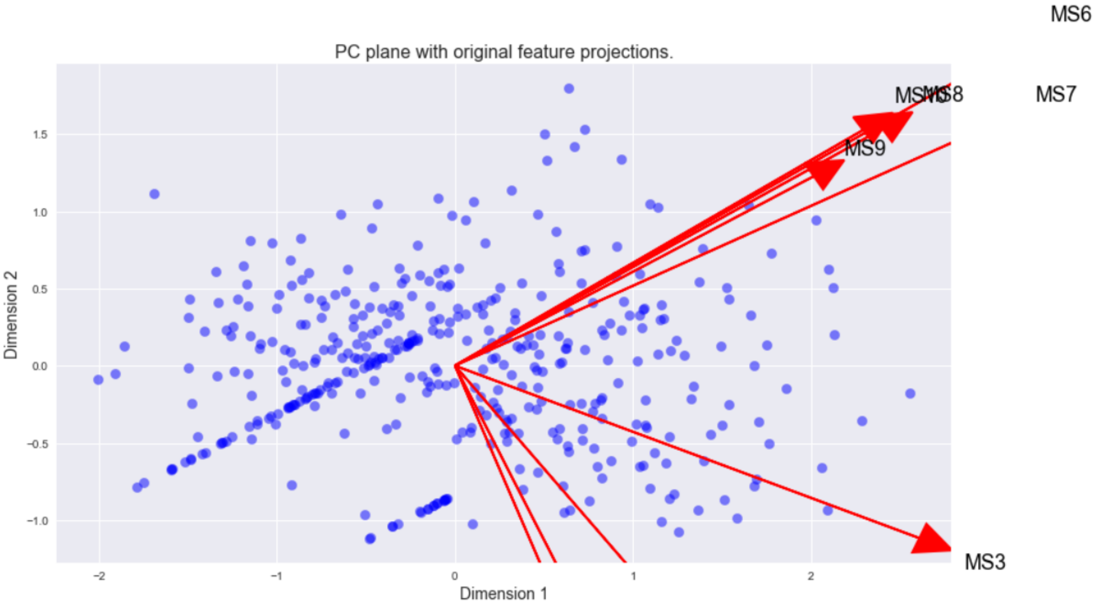
15 Next, I display the predictions of the sample data points and compare the results to see if the model predicts accurately.

The biggest challenge I ran into while doing this project is the data types and ensuring there were no NAN values after transformation.



If i look at the 1st and 2nd component 59.99% + 20.41% = 80.40% of the variance is explained. If we look at the first 4 components 59.99% + 20.41%+ 8.50% +4.10% = 93.00% of the variance is explained.

Then I ran a Biplot where each data point is represented by its scores with the principal components. The axes are the PC, here in this report it is the dimensions. The plot shows projects of the original features along with the components. This report helps discover relationships between the Principal components and the original features.



Refinement

In order to refine my code further and build a model that I could possibly use on larger database I further added 3 more parameters to my K-Means Clustering.

1st Parameter added – n_init = 25, default is 10. This is a parameter that tell the number of times K-means algorithm to run with different centroid seeds and then the final result is the best output of the consecutive run. In my case I set it to 25 runs.

2nd Parameter added = algorithm = “auto”
The classical is EM-style algorithm is “full”. However, I used “auto” because it chooses “elkan”which variation is more efficient because it uses the triangle inequality for dense and “full” for sparse data.

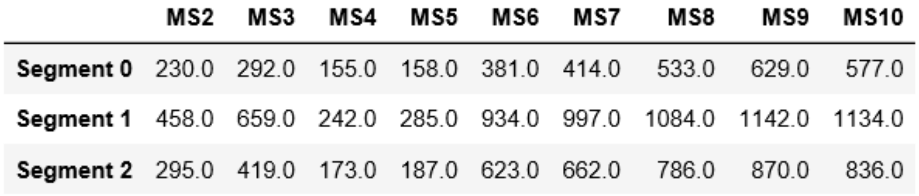
3rd Parameter added = init = “K-means++”
K-means++ selects initial cluster centers for k-mean clustering in a smart way to speed up convergence.

4th Parameter = verbose=1
Added this to step provide details on the steps and also for debugging.

Results

Model Evaluation and Validation

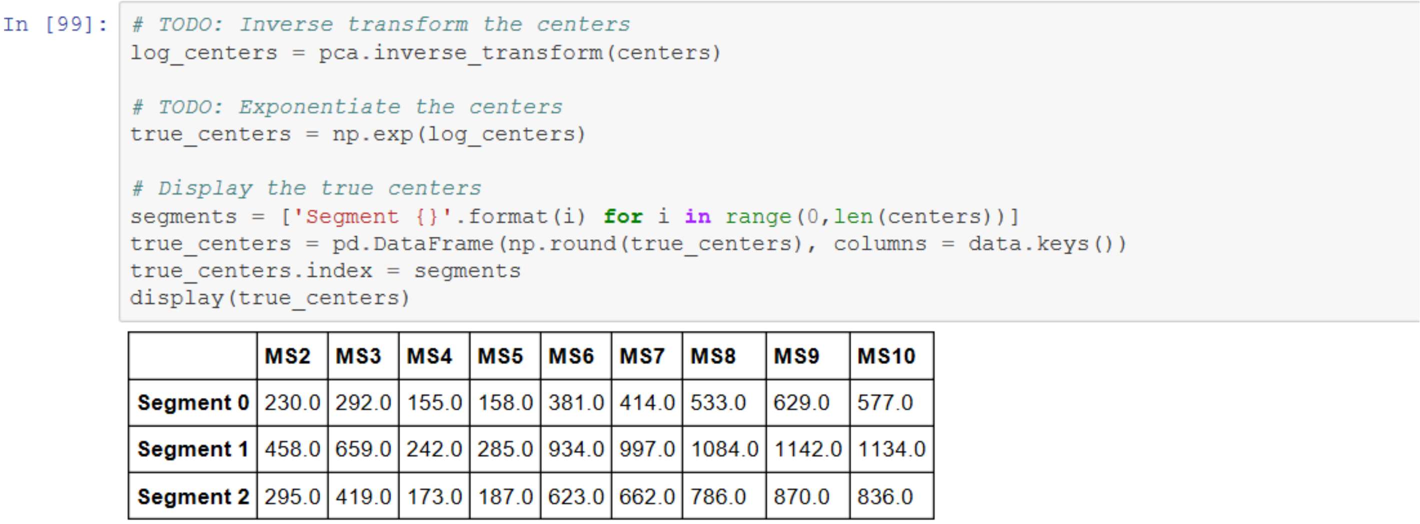
Data Recovery: I learned this feature in the program. Each cluster in the visualization has a central point. These are centers/means but not specifically data points rather they are averages of all the data points in the clusters. Since the data is reduced in dimension and scaled by a logarithm, we can recover the representative jurisdiction from these data points by applying the inverse transformations.



	MS2	MS3	MS4	MS5	MS6	MS7	MS8	MS9	MS10
Segment 0	230.0	292.0	155.0	158.0	381.0	414.0	533.0	629.0	577.0
Segment 1	458.0	659.0	242.0	285.0	934.0	997.0	1084.0	1142.0	1134.0
Segment 2	295.0	419.0	173.0	187.0	623.0	662.0	786.0	870.0	836.0

Being new to the concept of Machine Learning I relied heavily to the completed projects from Udacity as a guide and chose a model used in the unsupervised learning. I reviewed the feature relevance for all the milestones as one of the analysis. Example I looked at MS 6 to see if this milestone was providing any additional insight but it turned out that it didn’t as it was well correlated to few other Milestones. When I add MS1 it did change the clustering however, because MS1 Milestone is not related to execution but primarily related to administrative in office delays I ended up excluding it from the analysis. I trust the model performs accurately based on the predictions of the samples.

Justification



```
In [99]:  # TODO: Inverse transform the centers
          log_centers = pca.inverse_transform(centers)

          # TODO: Exponentiate the centers
          true_centers = np.exp(log_centers)

          # Display the true centers
          segments = ['Segment {}'.format(i) for i in range(0,len(centers))]
          true_centers = pd.DataFrame(np.round(true_centers), columns = data.keys())
          true_centers.index = segments
          display(true_centers)
```

	MS2	MS3	MS4	MS5	MS6	MS7	MS8	MS9	MS10
Segment 0	230.0	292.0	155.0	158.0	381.0	414.0	533.0	629.0	577.0
Segment 1	458.0	659.0	242.0	285.0	934.0	997.0	1084.0	1142.0	1134.0
Segment 2	295.0	419.0	173.0	187.0	623.0	662.0	786.0	870.0	836.0

These were my results!!!

Segment 0 cluster tells me that the site acq/jurisdiction approval and on-air time is well below 730 days/2 years. Perhaps these are candidates for consideration in reducing my delivery time.

Segment 1 cluster represent jurisdiction that will likely need to be considered for prefunding or increase in delivery time. The MS6 (Jurisdictional Approval) and MS10 (ON AIR) is way above the 730 days/2-year mark.

Segment 2 cluster tells me that we need not spend any time analyzing the strategy of funding on these. These can be funded as BAU efforts as we can expect them to be built in 2 years /730 days or close to it.

```
# Display the predictions
for i, pred in enumerate(sample_preds):
   print("Sample point", i, "predicted to be in Cluster", pred)
('Sample point', 0, 'predicted to be in Cluster', 1)
('Sample point', 1, 'predicted to be in Cluster', 2)
('Sample point', 2, 'predicted to be in Cluster', 1)
```

So i took the sample points and ran it against the predictions. I find that

Sample 0, ON AIR was 1035 days, the model accurately predicted this in cluster 1 which means we need to consider it for increased delivery date or prefunding.

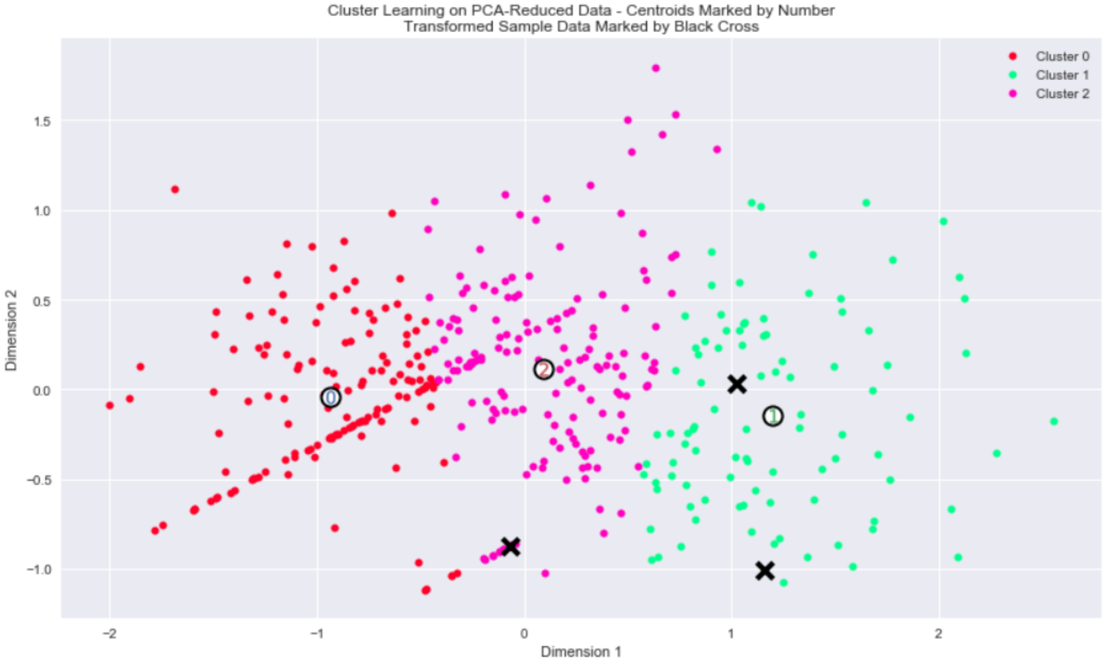
Sample 1 had an ON AIR of 661, which the model accurately predicted as BAU and

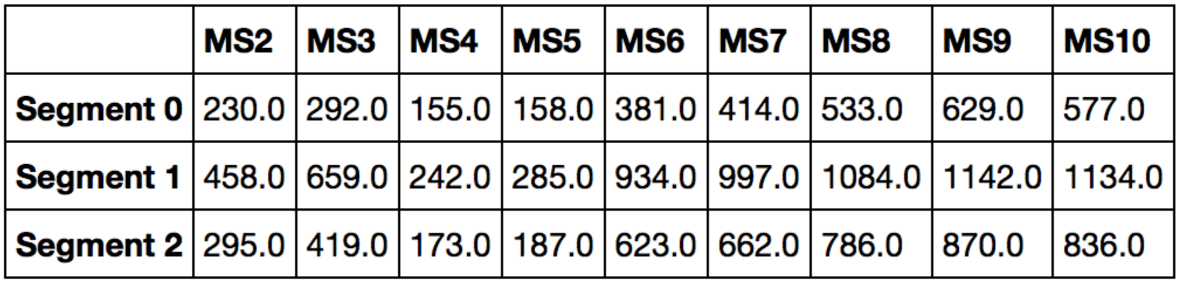
Sample 2 predicted cluster 1 which is also accurate because the ON AIR for it was 989 days this is well above 730 days and correctly classified under cluster 1, for review for prefunding.

I think the results are comparable to the Naïve Predictor performance. When I ran the Naïve predictor the accuracy score was 44.39%. and the Kmeans score is 43.25% for 2 clusters and 33.66% for 3 clusters. However, I feel better about the outcome instead of using the base model because the final model was refined, and the outliers were removed.

Conclusion

Free-Form Visualization





	MS2	MS3	MS4	MS5	MS6	MS7	MS8	MS9	MS10
Segment 0	230.0	292.0	155.0	158.0	381.0	414.0	533.0	629.0	577.0
Segment 1	458.0	659.0	242.0	285.0	934.0	997.0	1084.0	1142.0	1134.0
Segment 2	295.0	419.0	173.0	187.0	623.0	662.0	786.0	870.0	836.0

The key take away here is to observe the numbers in each milestones for the 3 clusters. Just to pick on one milestone lets review MS6. The center point for 1st cluster is 381 days, 2nd cluster is

934 days and 3rd cluster is 623 days. Clearly the 1st cluster is a candidate for funding reduction, cluster 2 is for increase in delivery date or prefunding and cluster 3 is a candidate for decrease in delivery date. Secondly if you look at the clusters you will see a clean segmentation, I don't see much blending of colors at the border. This visually gives me an impression that the model is trained well.

This is my first Machine Learning project done on a real-life problem. I believe i have successfully developed a model that has classified the data in 3 clusters. I tested my model via sample and it accurately predicts the correct cluster.
This information can be used to propose a change in funding strategy. When i first started the model i only wanted to find jurisdictions that will require prefunding or increase delivery time to complete the New Site but during the training process i learned that i can use this model to suggest the inverse. Meaning jurisdictions that predict to be in Segment 0, take less than 2 years to be build can be looked at to release funding with shorter interval time and possibly lower funding level. This can be savings for the company.

My base model accuracy score is 44.33% only looking at the ON AIR MS10 milestones and a larger dataset. My PCA model captures 93% of the variance after the outlier removals and gives a k means score of 43.25%. I chose 3 clusters although the score is 33.66% because i feel the dataset is richer.

Reflection

In this project I began with a data set of 579 records with 10 features. I reviewed and analyzed 9 features and
One thing I could possibly do is instead of removing all the outliers perhaps just remove the overlapping outliers. I found this project very similar to the customer segmenting project we did in the Machine Learning Program. I faced many challenges with completing this project but aside from which techniques to choose for so many techniques available out there the second struggle was dealing with the different data types and how some errors are prominent with Scikit libraries. Overall, I think the big take away for me is the preprocessing step and if the dataset is skewed to normalize it. Removing outliers and then using the clean data to predict seemed to be very powerful. One thing I learned during this process that if you have 0 values in the data you can replace the null values with means or averages. It's just a step I learned while running into issues with this fictitious dataset. After the data was scaled to a more normal distribution and outliers removed, I applied PCA to understand which dimensions of the data best maximize the variance of the features involved. I found this step powerful because it helped me reduce the dimensionality. Lastly, I created clusters. I used Silhouette coefficient for data points measures. I had learned this during the program as well that calculating the mean Silhouette coefficient provides a simple scoring method of a clustering. This helped me see the results of clustering and see how my samples I had selected predicted on those clusters. I am thrilled and very happy to see that the clustering worked, and the algorithm correctly identified the cluster based on the sample. This is an important project for me because the

algorithm I have created is scalable and can be used in real life within my organization. In fact, this model can be used on any classification project with features. With that said, this final model and solution fits my expectations for the given problem. I will be able to use this model real life data to request a funding strategy change in my organization.

Improvement

There are opportunities to improve. As mentioned earlier perhaps instead of removing all outliers I could only remove outliers that overlapped between the features (milestones).
I can try adding some additional minor milestones to see the performance.
From the algorithm standpoint i wanted to try the MLP regressor but didn't quite know how to implement it.
I am not sure if a better solution exists as I searched a lot to find a similar case online and didn't find anything of this nature on the cycle times for Telecommunication industry. With that said, if this was set as the new Benchmark it would be a good starting point.

References:

https://docs.scipy.org/doc/numpy/reference/generated/numpy.log.html
https://stackoverflow.com
https://matplotlib.org/examples/pylab_examples/multipage_pdf.html
https://classroom.udacity.com/nanodegrees/nd009/parts/5375cf82-14fe-422d-b737-7bc893e20a6d
https://onlinecourses.science.psu.edu/stat505/node/54 http://setosa.io/ev/principal-component-analysis/
https://www.dezyre.com/data-science-in-python-tutorial/principal-component-analysis-tutorial