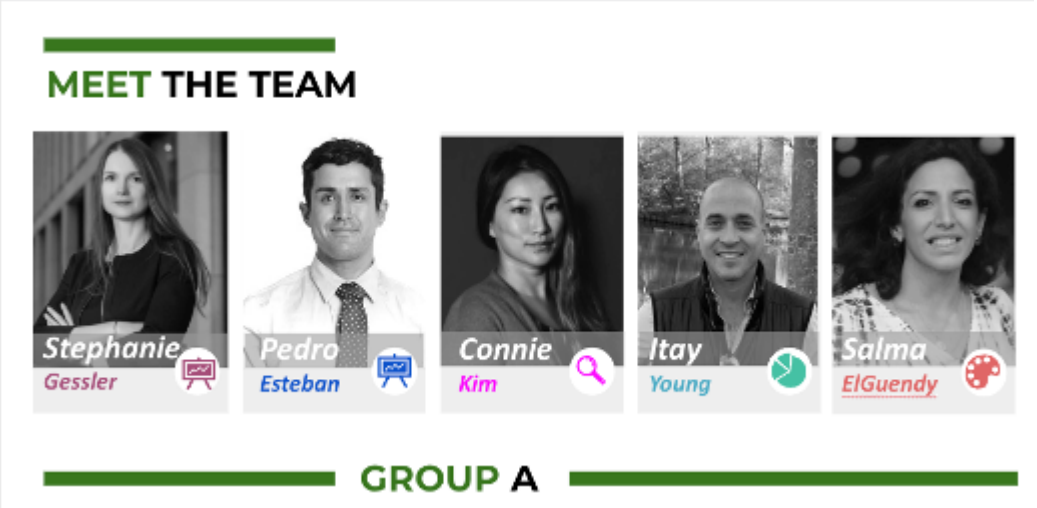
Machine Learning II Report - Group A



Introduction and Objectives 2

Description Overview of Notebook Files in Project: 2

Datasets Overview 3

Exploratory Data Analysis Findings 4

Data split 6

Creating Baseline Models for Baseline Performance 7

Baseline Model Results 8

Outliers Handling Analysis 9

Features Engineering 10

Feature Selection and Dimensionality Reduction 12

Models Comparison Table (Post Feature Engineering): 15

Further Explorations: Dimensionality Reductions with PCA 15

Best Model Assessment & Final Submission 17

Conclusions 18

General Lessons Learned 19

Appendices 19

Introduction and Objectives

Introduction:

Understanding forest composition is an important component for successfully managing our natural wilderness reserves, developing potential ecological related policies, and helping to determine necessary actions for preserving the health of forest reserves. Healthy forestry is important for sustainable wood production, fresh air, wildlife habitat, clean water, and quality of human life.

Objectives:

Our report has the objective of predicting tree cover types given specific known geologic and cartographic related data such as elevation, hydrologic, sunlight and/or soil data.

The area of interest includes four wilderness areas located in the Roosevelt National Forest of northern Colorado. Each observation is a 30m x 30m patch, and we have 565892 observations. The seven cover types are:

1. Spruce/Fir
2. Lodgepole Pine
3. Ponderosa Pine
4. Cottonwood/Willow
5. Aspen
6. Douglas-fir
7. Krummholz

The technical objective is to identify the best integer classification for the forest cover type out of our 7 known cover types, and to develop the best performing machine learning algorithm for this.

The data derives from Kaggle's hosted UCI repository source. The training set (15,120 observations) contains both features and the Cover_Type. The test set contains only the features. Our goal is to predict the Cover_Type for every row in the test set (565,892 observations) of these methods and techniques throughout this report.

Our team has completed a comprehensive exercise of Machine Learning techniques and explorations covered in this report, to achieve these goals.

Our hope is to deliver an algorithm product that can provide valuable information to scientists, field researchers, US Forest Service, and policy makers to help them understand Roosevelt Forest's composition better – ultimately to improve know-how for managing its health.

Description Overview of Notebook Files in Project:

Our overall project workflow was divided into 4 Jupyter Notebook files:

1. **Part 1** – Contains EDA analyses and Baseline models to create benchmarks
2. **Part 2** – Contains outliers analyses and handling, feature engineering, and our initial features selection methods
    a. Application of engineered selected features on our test dataset (forest_cover_type_test_data_feature_engineering)
3. **Part 3** - Contains run of Models based on initial features engineered and selected from Part 2 Notebook, testing, evaluations on initial results, and applying to test data. Evaluating **final**

results' improvements on performance, and applying to test data for final submission to see best cover type classifier

4. **Part 4** - Using feature selection from part 2 , experimenting with PCA and iteration of Models, hyper parameter tuning, evaluating final results' improvements on performance. None of the models in this iteration ended up being selected for final submission.
   Submission file : submission_consolidated.csv

[LINK TO FINAL PROJECT DELIVERABLE HERE]

Datasets Overview

The data set consists of 15,120 samples of 30m x 30m patches of forest located in northern Colorado's Roosevelt National Forest, for which the cover types are defined. And another dataset test of 565,892 samples that needs to be classified into seven cover types.

Dataset train

It is worth noting that 15,120 data points are split into 4 key wilderness areas and are based on forest with minimal human impact.

Key wilderness areas present in the data are:

1) Rawah Wilderness Area
2) Neota Wilderness Area
3) Comanche Peak Wilderness Area
4) Cache la Poudre Wilderness Area

The 15,120 samples dataset train also includes the target classifier data of the seven cover types. Here is an illustration for each cover type:

Data Fields for Dataset Train

We have 12 key elements impacting the cover type model in which wilderness areas are split into four and soil type is split into 40 different types as illustrated below.

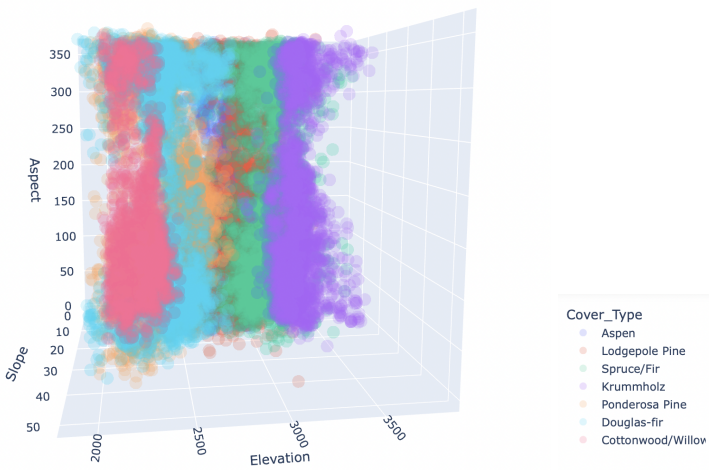
Name	Unit of Measurement	Description
Elevation	meters	Elevation
Aspect	degrees azimuth	Aspect
Slope	degrees	Slope
Horizontal_Distance_To_Hydrology	meters	Horz Dist to nearest surface water features
Vertical_Distance_To_Hydrology	meters	Vert Dist to nearest surface water features
Horizontal_Distance_To_Roadways	meters	Horz Dist to nearest roadway
Hillshade_9am	(0 to 255 index)	Hillshade index at 9am, summer solstice
Hillshade_Noon	(0 to 255 index)	Hillshade index at noon, summer solstice
Hillshade_3pm	(0 to 255 index)	Hillshade index at 3pm, summer solstice
Horizontal_Distance_To_Fire_Points	meters	Horz Dist to nearest wildfire ignition points
Wilderness_Area	(4 binary columns, 0 = absence or 1 = presence)	Wilderness area designation
Soil_Type	(40 binary columns, 0 = absence or 1 = presence)	Soil Type designation
Cover_Type	(7 types, integers 1 to 7)	Forest Cover Type designation

Exploratory Data Analysis Findings

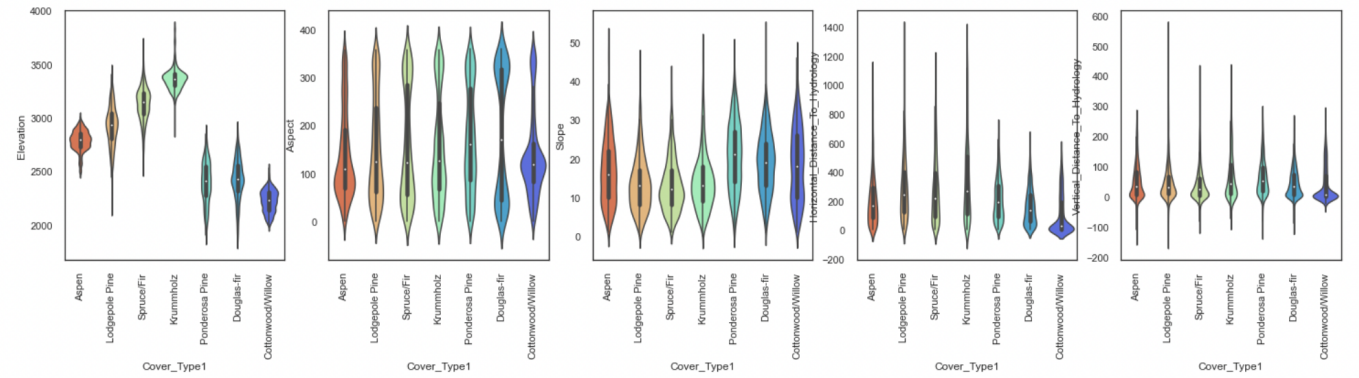
In this section we will provide our key analysis and findings with regards to the data-sets provided. Ancillary Jupyter notebooks will have more exploratory detail.

- **Data completeness:**
  - No missing values throughout the dataset samples (rows) and features (columns)
  - We confirmed that each sample occurrence has a single 'flag parameter' marked as one for wilderness area and soil type (by summing up all flag parameters and comparing it to the number of occurrences)
- **Parameter type:** All data is marked as integers, whereas flag / categorical parameters are already encoded with 0/1
- **Parameter distribution**: None of the features follow a normal distribution, which we confirmed visually and using the statistical Shapiro normality test
  - **Numerical variables** - not normally distributed
  - **Features that measure distance** - positively skewed
  - **Hillshade 9 am / Noon** - similar left-skewed
- **Parameter sign**: We noticed that vertical distance to hydrology includes negative values, where average altitude is ~1,500m;
- **Representation in Train and test datasets:**
  - Cover type are equally distributed and equally represented
  - Several soil types are unique to the different groups; Soil type 7 and 15 – no occurrences in the train data-set but included in the test data-set (105 and 3 occurrences);
  - Rare occurrences of soil types 8, 9, 28, 36, 25, 9 and 8 within the datasets
- **High correlation between parameters**, i.e:
  - Vertical distance to hydrology is correlated to: Horizontal distance to hydrology (0.65); Slope (0.27)
  - Elevation is correlated to: Horizontal distance to roadways (0.57); Horizontal distance to fire points (0.44); Horizontal distance to Hydrology (0.41); Wilderness area 3 (0.35) wilderness area 4 (-0.78)
  - Hillshade @ 9am is correlated to: soil type (0.24); hillshare @ 3pm (-0.78); Aspect (-0.59)
- **Elevation – Aspect – Slope features as interconnected classifiers**
  We observed these features on a 3D visual, where we saw that Elevation seems to be a key single feature for determining between Krummholz (purple), Spruce/Fir (green), Douglas-fir (light blue) and Cottonwood/Willow (pink) tree cover types without direct multidimensional relationship with the other parameters.

This is also supported in the below analysis, where only elevation seems to be directly linked with tree cover types



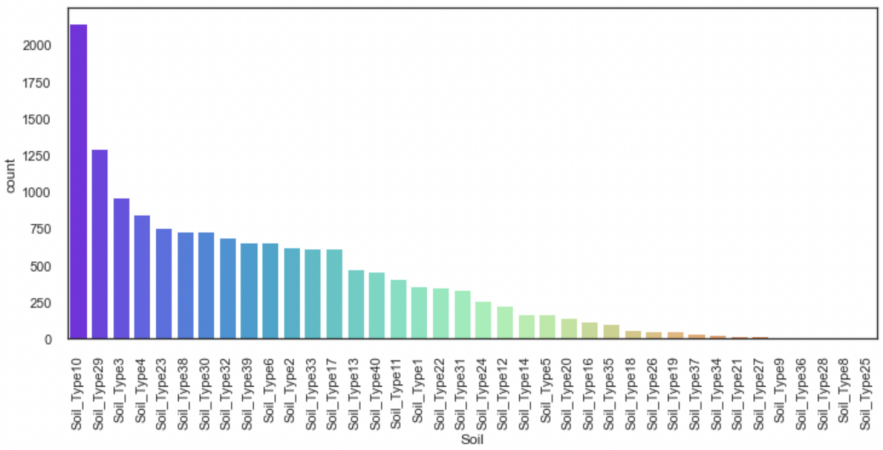
This is also supported in the below analysis, where only elevation seems to be directly linked with tree cover types



- Area types as tree cover classifiers: We checked whether area type has a direct link to tree cover type(s)

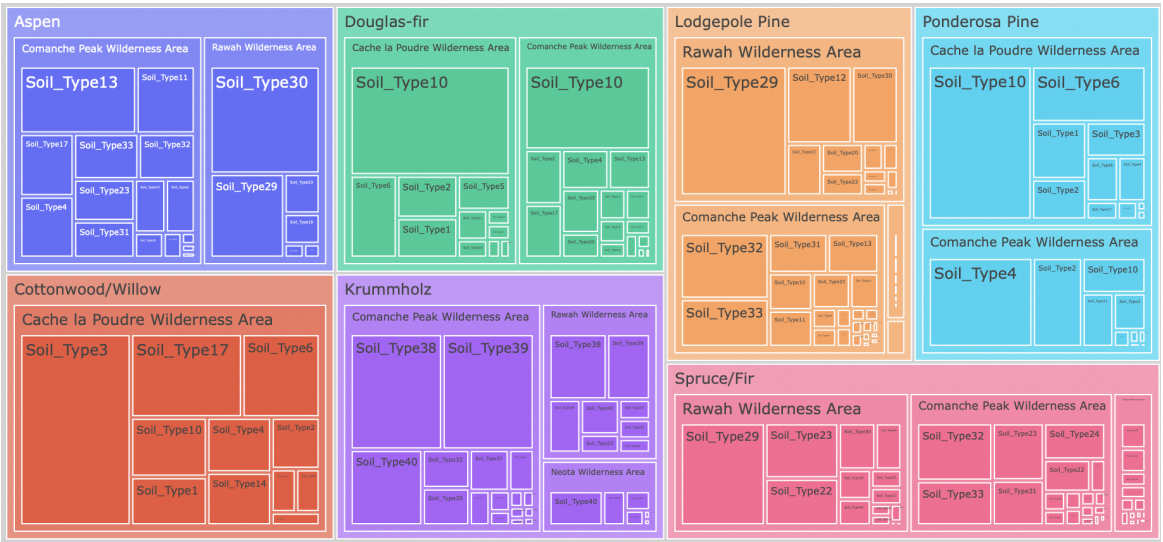
We observe that the tree cover type are not equally distributed across the wilderness areas, with the most even observations identified in Comanche Peak

- **Soil count inclusion:** We observe that we have 'too many' soil type features (40), which we would need to handle for the purpose of our analysis (e.g. binning)



We can see that many of the soil-type are under-represented within the data and as such, will be difficult to use them as classifiers without bucketing them and/or adding to them additional dimensions (combination with other features).

- Tree cover type – Wilderness area – Soil type relationship: We further observe that many of the soil-types are not directly correlated to certain cover types whereas some have more direct links (e.g. soil type 38 and Douglas-Fir), as represented below



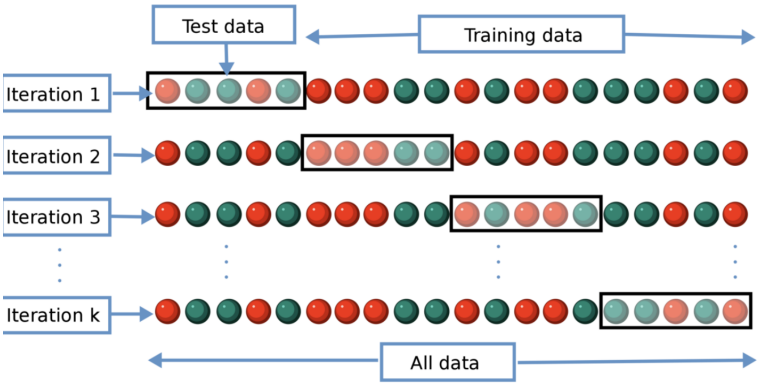
Data split

To be able to best train our model, we split our dataset into two groups:

- **Train dataset** – on which we will train our models
- **Validation dataset** – on which we will test and grade our final model results**.** We applied an additional validation split in our train dataset, resulting in train_train and train_val with an 80-20 split.

Since we saw in our previous EDA that the train set was not random and cover types were equally spread, we also stratified the split of the train into a train and validation set. Stratification allows us to have a balanced dataset with respect to the target variable.

Through iterative cross-validations, we generally wanted to test and check the performance in the split train data first, and to cross-validate between the train and test data.



[source: https://machinelearningmastery.com/train-test-split-for-evaluating-machine-learning-algorithms/

Creating Baseline Models for Baseline Performance

After reviewing our data - and before engaging in data cleaning, feature engineering, multiple model testing, hyperparameter tuning, dimensionality reduction, etc – we decided to use the 'raw' data to train and test a combination of models which will serve as our base line for the effectiveness of future model improvements.

To complete this task we first standardized our data using a standard scale (otherwise the algorithms will disproportionately weight features) and ensure that the distribution of cover types are well distributed between our subsets of data.

Models Tested – The models below are supervised learning algorithms/methods. We decided to go with these ones, as we believe they would be the best types to serve our purpose of trying to predict forest cover types.

1) **Random Forest:** Randomly creates and merges multiple decision trees into one "forest."
2) **Gradient Boost**: Tree based algorithm set to minimise the overall prediction error in subsequent iterations.
3) **Decision Trees:** Uses a set of rules to make decisions, similarly to how humans make decisions.
4) **K-Nearest Neighbours**: Stores all the available data and classifies a new data point based on the similarity then groups them together.
5) **Logistic Regression:** A statistical approach used to predict a dependent categorical target variable.
6) **Naive Bayes:** A probabilistic classifier using Bayes' theorem to classify

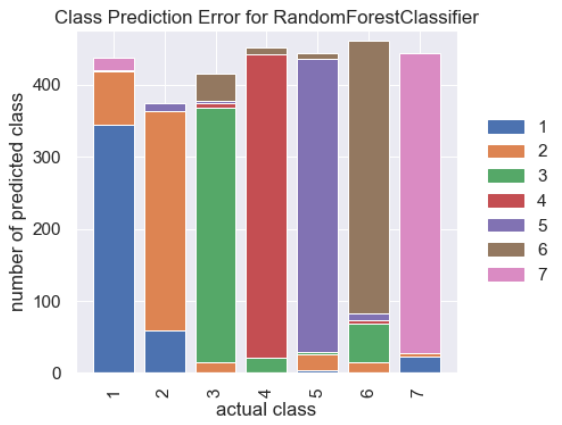
Using these 6 machine learning models to train and test our data, we reached the following results.

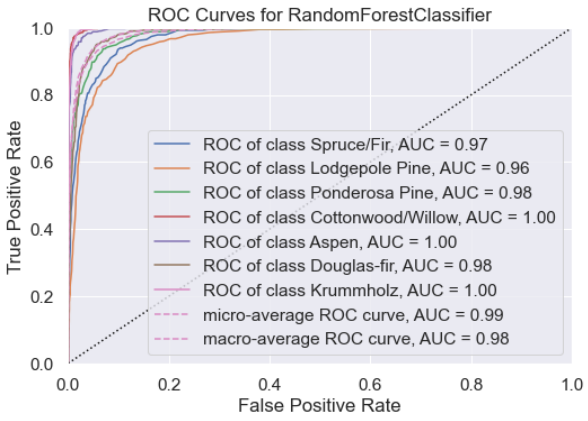
Baseline Model Results

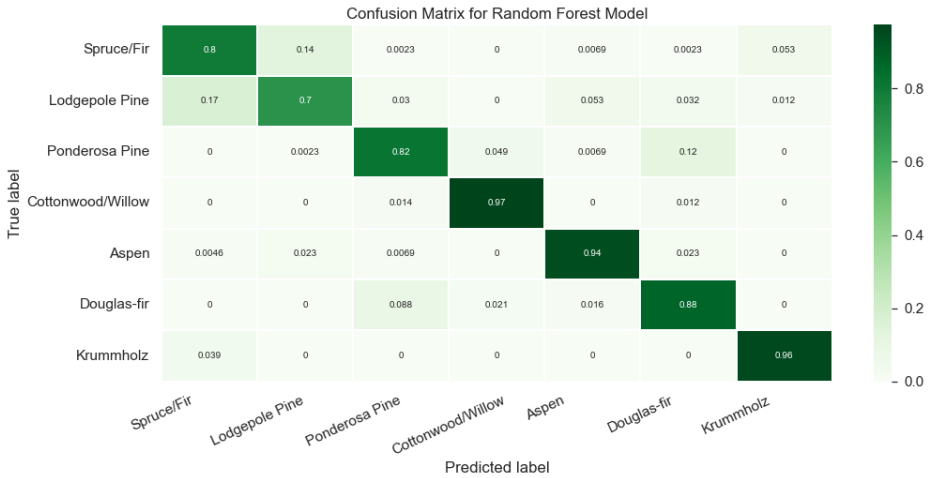
Baseline model = 6

	Random Forest	Gradient Boost	Decision Trees	K-NN	Logistic Regression	Naive Bayes
CV Accuracy	0.78	0.75	0.68	0.66	0.69	0.45
Accuracy	0.87	0.66	0.79	0.78	0.72	0.48
Precision	0.87	0.66	0.78	0.79	0.72	0.57
Recall	0.87	0.66	0.79	0.78	0.72	0.48
F1	0.87	0.66	0.78	0.78	0.72	0.39
score						
Assessment	*Best performing*	*Moderate score*	*Disqualified for overfitting*	*Good score*	*Good score*	*Lowest performing*

The above would suggest that the **Random Forest algorithm of 0.78** provided the best result. Further assessments on this model will show that it fairly well predicts most data types (0.82-0.97 accuracy) other than Lodgepole Pine and Spruce/Fir and that 2. its ROC curve for all tree cover type is fairly steep







Based on the graphics you can see that few cases are still misclassified. For instance class 1 (Spruce/Fir) is 14% of the time classified as class 2 (Lodgepole pine).The graph on the left shows and in the middle shows it visually, where the class gets classified. The Random Forest score will serve as our **baseline performance benchmark** in the next phases of our analysis.

Outliers Handling Analysis

Anomalies and outliers within our train data-set can negatively impact the accuracy of our certain types of predictive classification models, and as such, need to be evaluated to see how they should be handled. Also because our dataset is so large, we decided to test for multiple ranges of outliers.

We are well aware that it can be argued in some cases that outliers don't necessarily need to always be handled or removed from datasets. For example, if the dataset doesn't have true outliers (which are defined as problematic anomalies), and a population has natural variations, then it's possible that we don't necessarily want to remove these data points, if we don't need to.

Regardless, we wanted to assess and test the impact on performance of our best baseline model. We checked for IQR ranges, namely 4, 3, and 1.5 for potential outlier values – to be as thorough as possible.

For 4 IQR, we observed very minimal / insignificant outliers, as expected:
13 outliers in Vertical_Distance_To_Hydrology, 1 outliers in Hillshade_9am

For 3 IQR, we observed slight increases, as expected:
53 outliers in Horizontal_Distance_To_Hydrology, 49 outliers in Vertical_Distance_To_Hydrology, 3 outliers in Horizontal_Distance_To_Roadways, 132 outliers in Horizontal_Distance_To_Fire_Point

For 1.5 IQR, we observed the highest number of outliers, as expected:
There are 512 outliers in Horizontal_Distance_To_Hydrology, 586 outliers in Vertical_Distance_To_Hydrology, 830 outliers in Horizontal_Distance_To_Roadways, 645 outliers in Horizontal_Distance_To_Fire_Points

Without strong domain knowledge about geology or forestry characteristics, it can be a bit challenging to know whether these are true outliers versus just natural variations of these characteristics (features) in a dataset, in which case we should actually retain them to avoid excluding potentially important information.

To validate, we tested these anyhow to evaluate impact on our baseline model.

4 IQR trial Scores Below:

Algorithm	Accuracy	CV Accuracy	Accuracy with IQR4	CV Accuracy with IQR4
Random Forest	**0.87**	**0.78**	0.86	0.79

3 IQR trial Scores Below:

Algorithm	Accuracy Baseline	CV Accuracy Baseline	Accuracy with IQR3	CV Accuracy with IQR3
Random Forest	**0.87**	**0.78**	0.87	0.80

1.5 IQR trial Scores Below:

Algorithm	Accuracy Baseline	CV Accuracy Baseline	Accuracy with IQR3	CV Accuracy with IQR3
Random Forest	**0.87**	**0.78**	0.87	0.80

We can see that there was no significant improvement, only slightly but also evaluating the parameters in the test set. You can argue many of them are valid outliers. In addition, We also observed that datasets of our target variable of Cover Types got slightly imbalanced. So we ultimately decided not to drop the outliers in our datasets.

Features Engineering

After analysing the train data and researching the domain theory behind the parameters, we progressed to feature handling and engineering phase of our work. The objective of this phase is to enrich the train data of our model in multiple approaches such that the model will be able to better predict the result.

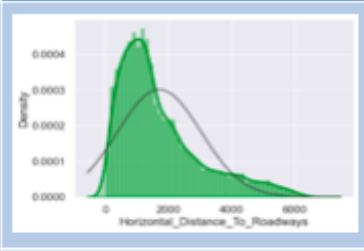
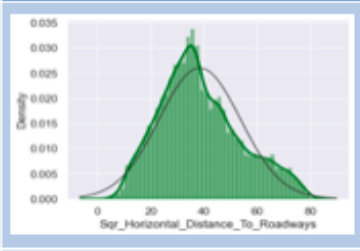
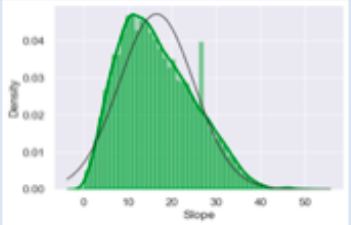
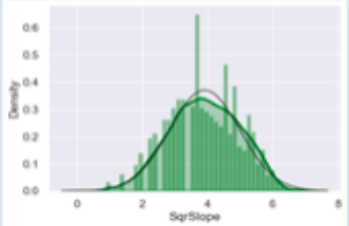
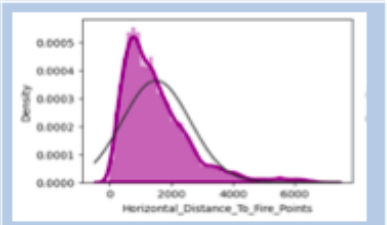
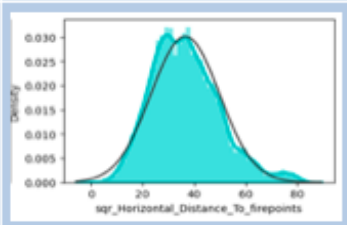
Bivariate Combinations

When studying our data we realized that many of the features may have interconnected relations and dimensions which can be useful to predict our classification problem. As such, we created new features using paired bivariate combinations of our predicting variables (excl. flag variables) using the ‘combinations’ method from itertools library. Provided the different scale of the features and the envisaged relationship, we decided to combine the parameters by way of multiplication.

Polynomial features
In this phase we decided to create new features by taking our existing features and increase them in power (i.e. [Feature] ^ ^{2}). We hope that this new scale of features can later improve our model by introducing a new relationship / dimension. We note that this was applied to all features including the Bivariate Combinations excluding the flag features.

Transformation of existing features
In our EDA analysis we observed that many of the features carry a positive or negative skew. We therefore decided to perform logarithmic , square root and in some cases a box cox transformation to the data, targeting to reach a more normalised distribution which will enrich the dimensionality of the data for our model training.

An example to this would be the following:

Parameter	Before transformation	After transformation
Horizontal Distance to Roadways (negative skew adjusted via Square Root transformation)		
Slope (negative skew adjusted via Square Root transformation)		
Horizontal Distance to Firepoints (negative skew adjusted via Square Root transformation)		

Mathematical and geometrical combination of existing features
In this section we introduce new features by implementing mathematical or geometrical relationships of *adjacent* features seeking to best illustrate their relationships. We hope that this combination of similar datapoints will allow us to better train the model

Vertical Distance to Hydrology
We combined the two correlated features 'Vertical Distance To Hydrology' and 'Horizontal Distance To Hydrology' into a single feature 'Distance To Hydrology' using the Pythagoras Theorem for distance calculation. As the data is positively skewed we performed various log and SQR transformations ultimately deciding to use SQR.

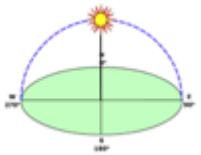
Mean Hillshade
We introduced a new feature which consists of the mean average of 3 hill shade data points (features) through different time of the day: 9am; 12pm and 3pm; The distribution of this feature was negatively

skewed, and as such we normalised it using BoxCox Transformation. Further, we used Hillshade ratios divided by the index of 255 and introduced a new feature Hillshade all day which bin all Hillshade variables into above 150 index with 1 and below with zero. In this way you can see where there is more hillshade exposure.

Features utilizing Domain Data

Aspect flag features

Aspect represents the angular direction to the sun, measured from north clockwise in degrees from 0 to 360. As such we used this numerical feature and transformed it into 4 flag features: Aspect_North (315 to 45 deg); Aspect_East (45 to 135 deg); Aspect_South (135 to 225 deg); and Aspect_West (225 to 315 deg);



Binned elevation

We discrete the elevation variables into intervals of 50.

Addition and subtraction of features with the same scale

Since addition and subtraction gave us a bad performance using for loop, we integrate only the variables which has the same scale like meters. We also used the SQR and evaluated the performance of these variables.

Soil types: Geo-climate grouping

In our EDA chapter we address our concern over the model's ability to properly use the soil features provided that: (i) there are 40 soil type features and only 7 tree cover type classifications; (ii) the absence or poor representation of soil types in our training data-set; and (iii) the unbalanced representation of certain soil-types between the training and testing data-sets.

Thus, we introduced the following soil related features utilizing domain data whilst reducing the number of soil linked flag features / parameters. We hope that these features will allow the model to better utilize and interpret the soil type related features and dimensionality in the classification algorithms.

Thus, we introduced the following soil related features utilizing domain data whilst reducing the number of soil linked flag features / parameters. We hope that these features will allow the model to better utilize and interpret the soil type related features and dimensionality in the classification algorithm.

Climatic feature engineering to group soils

The USFD - an American federal agency for forest service dependent on the department of agriculture, classified soil types according to climatic zone (first digit) and geology (second digit). On this basis we created 2 flag features in our model:

- **Climate grouping** - includes 7 categories: Lower Montane Climate; Montane Dry Climate; Montane Climate; Montane Dry and Montane Climate; Montante and Subalpine Climate; Subalpine Climate; and Alpine Climate
- **Geology grouping** – includes 4 categories: Alluvium Soil; Glacial Soil; Mixed Sedimentary Soil; and Igneus and Metamorphic Soil;

Soil feature engineering by family types

In this exercise we grouped the soil types into soil families, reducing the 40 features into 16 groups. The new soil type groups (flag variables) are as follows: Cathedral (1); Retake (2, 4); Vanet (5); Wetmore (6); Gothic (7); Limber (8); Troutville (9); Legault (12, 29); Gateview (17); Rogert (18); Como (30); Bross (36); Catamount (10, 11, 13, 26, 32, 31, 33); Leighcan (21, 22, 23, 24, 25, 27, 28); Moran (38, 39, 40); Others (3, 15, 37, 34, 35, 20, 14, 16, 19) }

Soil feature engineering by stoniness type

Lastly, we suspect that the stoniness level of the soil can be a dominant classification dimension in explaining the tree cover type of an area. As such we grouped the soil types into 3 stoniness flag variables: Stony (22 types); Rubbly (6 types); and Others (12 types);

At this point of our work – having developed our features for parameter distribution, feature combinations, dimensionality enhancement, mathematic / geometrical combinations, domain knowledge interpretation / grouping, etc., we feel comfortable that we can progress with our modelling workflow and engage with feature selection. By this time, the dataset consists of 108 columns vs. 56 at the outset reflecting the new features which we inserted into the database.

Feature Selection and Dimensionality Reduction

At this point of our analysis – having significantly enriched our feature 'pool', we wish to start 'narrowing the net' eliminating features which may negatively affect the result and/or overfit our model.

Numeric feature standardization

Our train data-set consists of many features which are represented in different units and scales, e.g. some in distance, degrees, square roots, box cox, etc. These differing scales can negatively impact the algorithm analysis of the train data such that it will overweight features which are represented in a higher vs. lower numeric scale even where the feature predictive importance is identical or lower. This can especially occur in non-tree based algorithms. To avoid this we applied standard scaling to all numeric parameters.

Dropping redundant features

In the previous section of this report we have explored the development of new features in the data-set. With the completion of this exercise, we reach feature redundancies which we now want to eliminate. As such, we decide to eliminate the following features: the 40 soil type Boolean features, which were grouped and categorized in alternative methods; soil type features by family and complex, as we feel that our Geological and Climate grouping will bring more domain meaning to our analysis.

Dropping highly correlated features

We notice that some of our features are highly correlated to each other. This derives from both intrinsic correlation between the features but also provided our feature engineering exercise where we conducted various feature combinations. This correlation between the explanatory features can negatively effect the results of some of the models which we wish to employ on our dataset (e.g. logistic regression) such that the model will allocate too much weight to these correlated features when predicting the outcome. We are well aware that elevation is a key feature in our data which is both highly correlated to other features and has significant representation in feature combinations. After several iterations we decided to remove **some** of the elevation related features to ensure that the high weight of elevation will not negatively impact the other features when training our final model.

Feature selection methodology

At this final point we wish to reduce the number of features that we use to train our algorithms such that we will use those features that carry the best prediction / classification information (correlation to our target variable). We are also aware that some features can provide marginal positive explanatory contribution however cause our model to overfit. As such we reflect this risk by 'penalizing' the model for incremental features used.

In order to shortlist the features to be used in our train dataset, we applied various feature selection algorithms and regularization methods that would help us with selecting features to use. We started with using only **46 features** for each selection method. We used **Lasso Regularization, Anova F-value, Pearson Correlation, Recursive Feature Elimination(RFE)**, **Random Forest Classifier, XGboost** and **Extra Tree Classifier**, to indicate signal on which features to select, and then proceeded to use the best of all selections. A thorough technical explanation for each approach utilized, are provided in Notebooks. The results on features selections are presented below:

	Feature	ExtraTree	Pearson	RFE	Anova	Logistics	Random Forest	XGB	Total
1	Wilderness_Area4	True	True	True	True	True	True	True	7
2	Subalpine_Climate	True	True	True	True	True	True	True	7
3	Alpine_Climate	True	True	True	True	True	True	True	7
4	sqr_Road-Hydro	True	True	True	True	True	True	False	6
5	sqr_Road-Fire	True	True	True	True	True	True	False	6
6	sqr_Road+Fire	True	True	True	True	False	True	True	6
7	binned_elevation	True	False	True	True	True	True	True	6
8	Wilderness_Area3	True	True	True	True	True	False	True	6
9	Wilderness_Area1	True	True	True	True	True	False	True	6
10	Montane_Climate	True	True	True	True	True	False	True	6
11	Lower_Montane_Climate	True	False	True	True	True	True	True	6
12	Horizontal_Distance_To_Roadways_x_Horizontal_D...	True	True	True	True	True	True	False	6
13	sqr_Road+Fire+Hydro	True	True	True	True	False	True	False	5
14	sqr_Ele-hydro	True	False	True	True	True	True	False	5
15	sqr_Ele+road	True	True	True	True	False	True	False	5
16	sqr_Ele+hydro	True	False	True	True	True	True	False	5
17	sqr_Ele+fire	True	False	True	True	True	True	False	5
18	sqr_Ele+Road+Fire+Hydro	True	True	True	True	False	True	False	5
19	Horizontal_Distance_To_Roadways^2	True	True	True	True	False	True	False	5
20	Hillshade_9am_x_Hillshade_Noon	False	False	True	True	True	True	True	5
21	Elevation_x_Hillshade_Noon	True	False	True	True	True	True	False	5
22	Elevation_x_Hillshade_9am	True	False	True	True	True	True	False	5
23	sqr_Road+Hydro	False	True	True	True	False	True	False	4
24	sqr_Ele-road	True	False	True	False	True	True	False	4
25	sqr_Ele-fire	True	False	True	False	True	True	False	4
26	sqr_Distance_To_Hydrology	True	False	True	True	True	False	False	4
27	Sqr_Horizontal_Distance_To_Roadways	False	True	True	True	False	True	False	4
28	Igneus_and_Metamorphic_Soil	False	True	True	False	True	False	True	4
29	Horizontal_Distance_To_Roadways_x_Hillshade_Noon	False	True	True	True	False	True	False	4
30	Horizontal_Distance_To_Roadways_x_Hillshade_9am	False	True	True	True	False	True	False	4
31	Horizontal_Distance_To_Hydrology_x_Hillshade_3pm	False	False	True	True	False	True	True	4
32	Hillshade_Noon_x_Horizontal_Distance_To_Fire_P...	False	True	True	True	True	False	False	4
33	Hillshade_9am_boxcox	True	False	True	False	True	True	False	4
34	Elevation_x_Horizontal_Distance_To_Roadways	False	True	True	True	False	True	False	4
35	Elevation_x_Horizontal_Distance_To_Hydrology	False	False	True	True	False	True	True	4

In our final selected feature data-set we included those features that were selected at least **4 times** as best predictors by the 7 feature selection algorithms mentioned above

Summary of selected features

Having selected our train data features, we review these 35 and attempted to bucket into categories or themes, to try to make sense of the newly engineered and selected features:

Total final # of features selected (post feature selection methods) = 35

Area related	Climate related	Horizontal distance related	Square	Square Distance related	Elevation related	Soil types related	Hillshade related
Wilderness_Area3	Subalpine_Climate	Horizontal_Distance_To_Roadways_x_Horizontal_Distance_To_Fire_Points	sqr_Road+Fire+Hydro	sqr_Distance_To_Hydrology	Elevation_x_Hillshade_Noon	Igneus_and_Metamorphic_Soil	Hillshade_Noon_x_Horizontal_Distance_To_Fire_Points
Wilderness_Area1	Alpine_Climate	Horizontal_Distance_To_Roadways^2	sqr_Ele-hydro	Sqr_Horizontal_Distance_To_Roadways	Elevation_x_Hillshade_9am		Hillshade_9am_boxcox
[Wilderness_Area4	Montane_Climate	Horizontal_Distance_To_Roadways_x_Hillshade_Noon	sqr_Ele+road		Elevation_x_Horizontal_Distance_To_Roadways		Hillshade_9am_x_Hillshade_Noon
	Lower_Montane_Climate	Horizontal_Distance_To_Roadways_x_Hillshade_9am	sqr_Ele+fire		Elevation_x_Horizontal_Distance_To_Hydrology]		Hillshade_9am_x_Hillshade_Noon
		Horizontal_Distance_To_Hydrology_x_Hillshade_3pm	sqr_Ele+Road+Fire+Hydro		binned_elevation		
			sqr_Road+Fire+Hydro				
			sqr_Road-Hydro				
			sqr_Road-Fire				
			sqr_Road+Fire				
			sqr_Ele-road				
			sqr_Ele-fire				

We can observe that:

- The features which we incorporated in the feature engineering phase represent the vast majority of the features ultimately selected
- We are pleased to see that the selected combination of features as selected by the models cross-checks with our domain knowledge - i.e. balances elements such as Wilderness Area, Climate, Distance, Elevation, Soil Type, Hillshade; all of which we expect to be key for predicting the tree cover type
- 12 features - which include wilderness, climate, and distance metrics, were selected by 6/7 of the models
- Elevation - which was identified as a high importance feature in our EDA, has 5 representations
- In the Jupyter notebook, you can observe the distribution of feature importance for each of the models deployed - we note that some features such as Alpine / Subalpine Climate, Binned Elevation, Wilderness Area 4 / 1, Glacial Soil and others seem to have a bigger impact and correlation on the prediction vs. some of the others selected

Having selected our features, we progressed to training our model based on these.

MODEL TRAINING and HYPERTUNING

Starting with data splitting with selected features and standardised them. We do ignore the fact that some models cannot handle multicollinearity because these model would needed a complete separate selection process. In order to compare all classifiers performances, we used the same feature set. However, in the correlation heatmap you can see highly correlated features.

For each of the above models previously used on baseline dataset, we re-ran the model on the cleaned dataset using the selected features, examined the overall score of the model (accuracy),

illustrated the findings in confusion matrix format as well as class prediction graphs, ROC, F1 score, precision, and recall.

At this point, we decide to focus on CV Accuracy as our key performance metric for predicting forest cover types, and this would be our determining factor for what to eliminate, and what to move forward with.

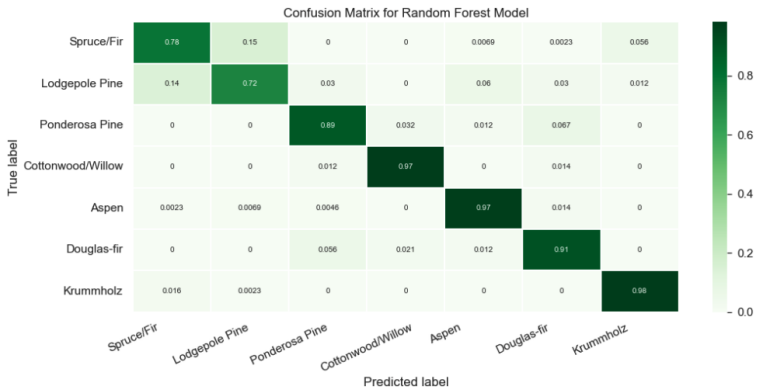
Models Comparison Table (Post Feature Engineering):

		Tree Based Models									
	Baseline Random Forest	Random Forest	XGBoost	Decision Trees	Extra Tree Classifier	Naive Bayes	SVM	K-NN	Logistic Regression	Voting Classifer	Voting Classifer
Val CV Score Mean	0.78	0.79	0.79	0.72	0.816	0.57	0.78	0.78	0.66	0.819	0.819
Train CV Score Mean	0.85	0.86	0.86	0.75	0.88	0.56	0.85	0.85	0.69	0.87	0.87
score											
Assessment	*Best performing*	*Good score*	*Good score*	*Moderate score*	*Good score*	*Lowest performing*	*Good score*	*Good score*	*Lowest performing*	*Best performing*	*Selected Model*

From the models' scores, we concluded that Voting Classifier is the best performing model since it returns highest scores on Val CV mean score at 0.819. As you can see in the above comparison table, we kept the baseline best model as a benchmark against all models scores. Also, we grouped first all tree based models as they are usually best at classification with wide space of features followed by the rest of the models such as Naive Bayes, KNN, SVM, and Logistic Regression. The Voting Classifier model here was expected to outperform the others since it takes into consideration all of the previous models and optimises from there.

Below are some further insights for the Voting Classifier model:

The 1st graph here shows how each predicted type is well represented within its bar, while the confusion matrix graph also draws a strong correlation between the features using this model.

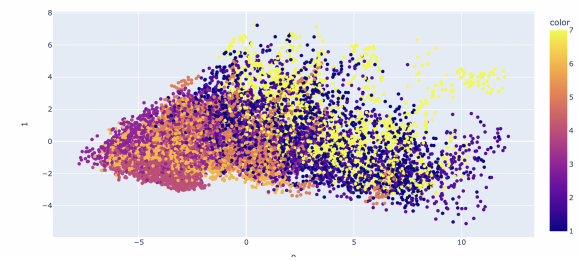


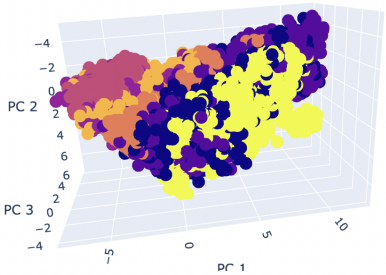
Further Exploration with Dimensionality Reduction Methods

To further explore if we could improve our models even further – we experimented with dimensionality reduction work, with PCA method. Principal Component Analysis (PCA) enables the transformation of any high number of variables into a smaller number of uncorrelated variables called principal components (PCs), developed to capture as much of the data's variance as possible.

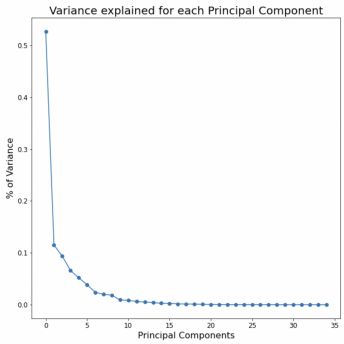
Once the PCA method is applied on the selected 35 features reached in the feature selection process above, we re-run the selected models again. We hope that the PCA will enable us to improve the model performance as (i) we acknowledge that some of the features are highly correlated, which can negatively impact the performance of some (non tree) algorithms; and (ii) we have sufficient number of features (35), which we consider to still be high and complex; therefore we wanted to apply this technique to reduce scope a bit and re-assess the scores. In normal circumstances, we are also aware that in theory the application of PCA may also be doable in pre-processing steps to support initial feature selection.

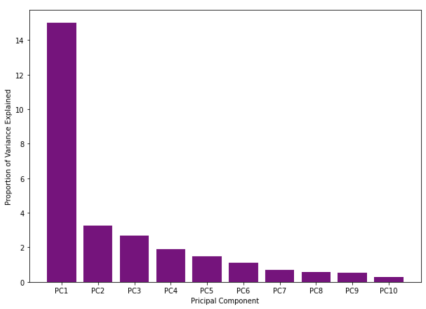
Using the first 2 principle components show that the data with 2 PCAs is very well differentiated as shown in the 3D graph below. With only 2 PCA or 3PCA you can already see that the cover types can be separated from each other.

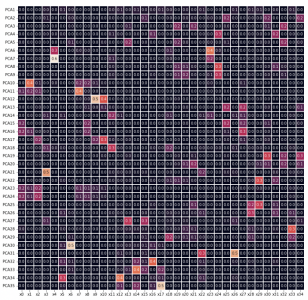




To illustrate the impact using the 2 PCA method, the 1st graph shows the variance explained to each principal component, the 2nd graph demonstrates the variance in percentages with bars, and we also created a heatmap to showcase the correlation between them.







Applying the PCA method returned an optimised features set of 12 different features which gives us the optimum CV score as per the grid graph below. Hence, we decided to proceed with the 12 selected features for the best model assessment. After testing with PCA, however, we decided we would not ultimately apply this to our final model for production, due to lack of significant performance improvements.

At this stage, we proceeded with 4 different models to rerun with the 12 features based on the PCA outcome as part of dimensionality reduction; 1) Random Forest, 2) XGBoost, 3)

Extra Trees Classifier, and 4) Ensemble Methods. We realized that PCA did not improve the models performance, and we concluded that Voting Classifier model is still the best performing model reached after the feature engineering section

Best Model Assessment & Final Submission

As already mentioned the best performing model reached after the feature engineering section is the ensemble method Voting Classifier.

	Tree Based Models										
	Baseline Random Forest	Random Forest	XGBoost	Decision Trees	Extra Tree Classifier	Naive Bayes	SVM	K-NN	Logistic Regression	Voting Classifer	Voting Classifer
Val CV Score Mean	0.78	0.79	0.79	0.72	0.816	0.57	0.78	0.78	0.66	0.819	0.819
Train CV Score Mean	0.85	0.86	0.86	0.75	0.88	0.56	0.85	0.85	0.69	0.87	0.87
score											
Assessment	*Best performing*	*Good score*	*Good score*	*Moderate score*	*Good score*	*Lowest performing*	*Good score*	*Good score*	*Lowest performing*	*Best performing*	*Selected Model*

The best scoring model from the above based on the **Val mean CV score is the Voting Classifier with 0.819**.

Conclusions

The appended diagram (appendix 1) illustrates our recommended workflow procedure for predicting tree cover types in the Roosevelt National Forest of northern Colorado. We started by splitting the dataset into two parallel paths. In our feature engineering and selection process, we ended up with 35 selected features incorporating key underlying information such as Wilderness Area, Climate, Distance, Elevation, Soil Type, Hillshade (transformed) data.

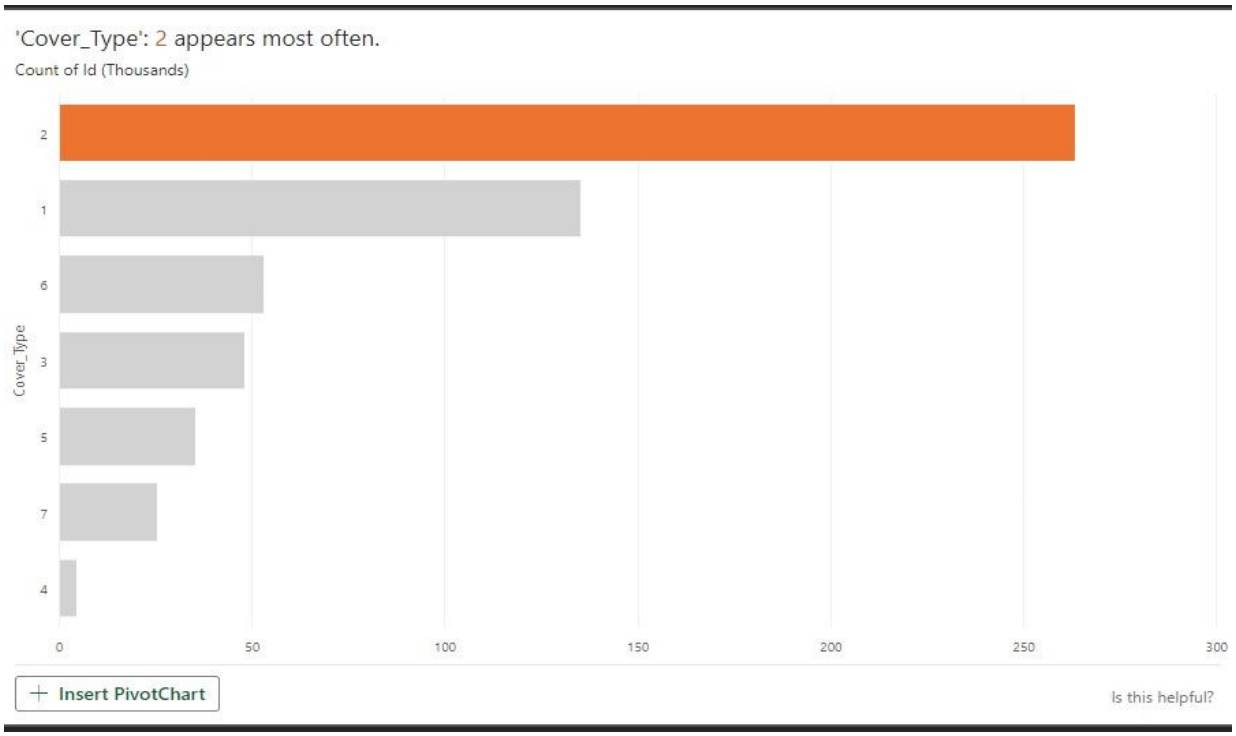
Firstly was where we directly applied the models with no transformations to the dataset to find baseline models scores, and secondly was where we applied features engineering/ selection work, and thirdly was where we explored other dimensionality reduction methods. We kept rerunning the models at each of these different stages to test and to compare the models scores to the baseline models scores.

This approach enabled us to better assess the classification performance and to select the best performing model with CV scores Vs. training scores as well as avoid overfitting syndrome, to apply to our test set for

Generally we expected that Decision Tree related algorithms would most likely perform better for our use case, compared to binary classification types like logistic regression for example which may be limited when handling a larger and more complex features space.

After adding some **Ensemble Methods,** such as **Extra Tree Classifier** and **Voting Classifier**, we ultimately found that these resulting types of models added more predictive strength increasingly with each model iteration. Because these model types rely and build upon pre-existing models' performances, ensemble-based models tend to have higher predictive accuracy, compared to predecessor individual models.

Having completed our analysis on the train data, we implemented our methodology on the test data provided and are pleased to share with you our below results on the 565892 observations provided:



Key: 1. Spruce/Fir; 2. Lodgepole Pine; 3. Ponderosa Pine; 4. Cottonwood/Willow; 5. Aspen; 6. Douglas-fir; 7. Krummholz;

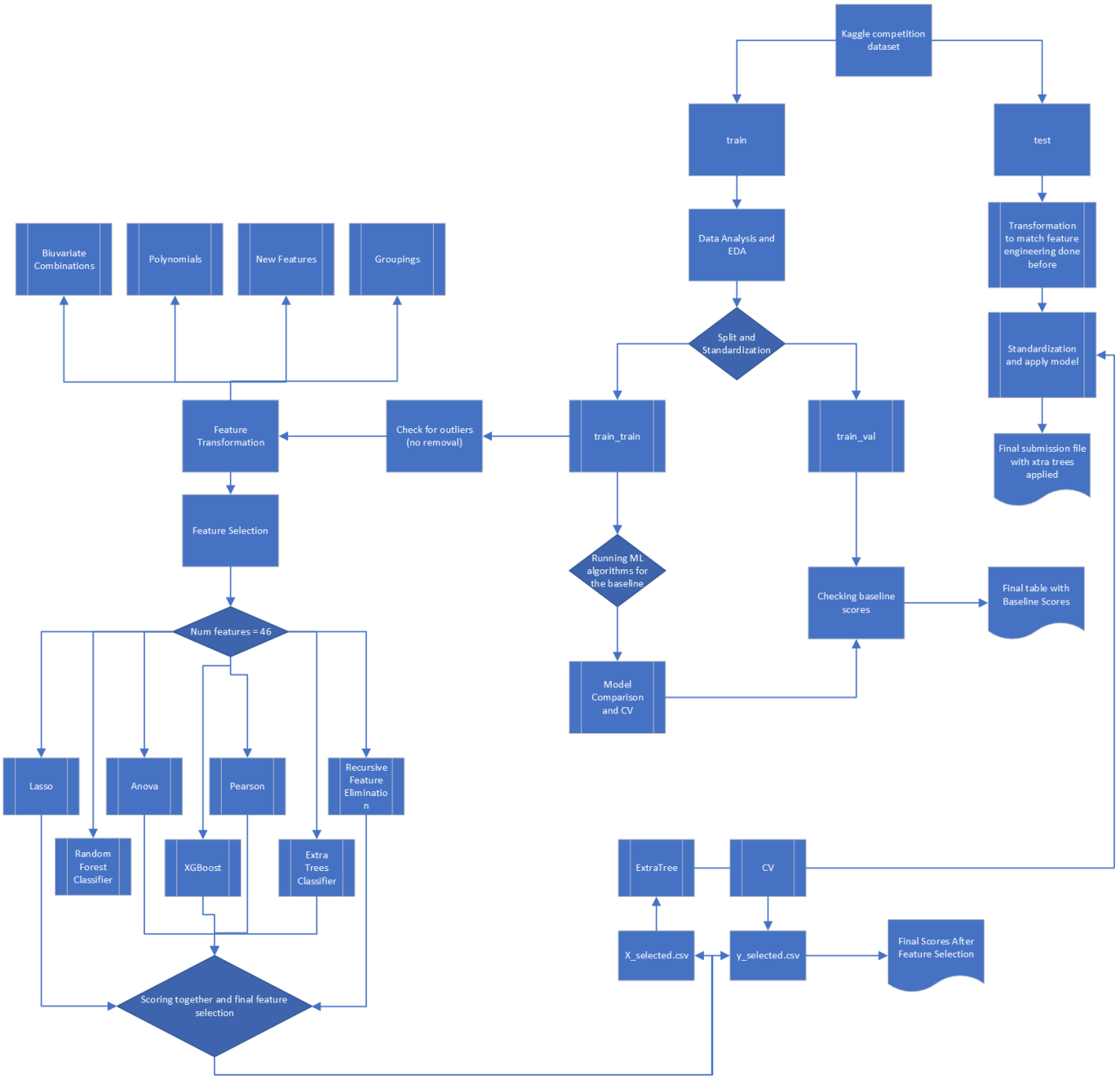
As can be observed, we expect that the majority of the tree cover type in the test data-set is Lodgepole Pine (#2) and Spruce/Fir (#1).

We hope that this insight can provide valuable information to scientists, field researchers, US Forest Service, and policy makers to better plan and execute their important work.

General Lessons Learned

We can apply our lessons learned and recommendations in themes:

- **Apply appropriate methods at best practice step**s.
    - Scaling can be very important at early stages to ensure better feature selection as well as to have a better performing model
- **Test, test, test – to validate what we *think we know*.**
    - Outlier removal does not necessarily improve model performance, and can even be detrimental. Testing helped us to confirm that removing was in fact not beneficial after all, despite our assumptions.
    - PCA did not improve either models' performance, despite our assumptions.
    - For hyper parameter tuning, it can be impactful for some models but not all of them (as shown in the best model section) while it didn't have much impact on the rest of the models.
- **Don't forget to handle Validation test sets. And to Cross-Validate.**
    - Making sure that our Validation data reflects the same characteristics as Training data is crucial to ensuring quality and validity of our model results.
    - Cross-validating is crucial for giving the *best realistic signal* for how accurately a predictive model performs in actual practice.
- **Simplify. Focus on what matters.**
    - It's not necessary to keep all techniques applied, which may or may not work. It can be better to trim out unnecessary steps, if they do not improve performance. We should focus on where we see the right signals, to fine-tune the pipeline and model in order to get to the best overall final product.
- **More experience and domain knowledge is incredibly helpful**.
    - Having more experience accumulates knowledge about which algorithms may tend to work best, for which industries and domain use cases.
    - We can only make assumptions based on theories otherwise, but can only validate our assumptions by simply practicing, experimenting, observing, and iterating, to accumulate more best practice knowledge through experience.

Appendices

Appendix 1 of [final simplified workflow proposed for production](#):