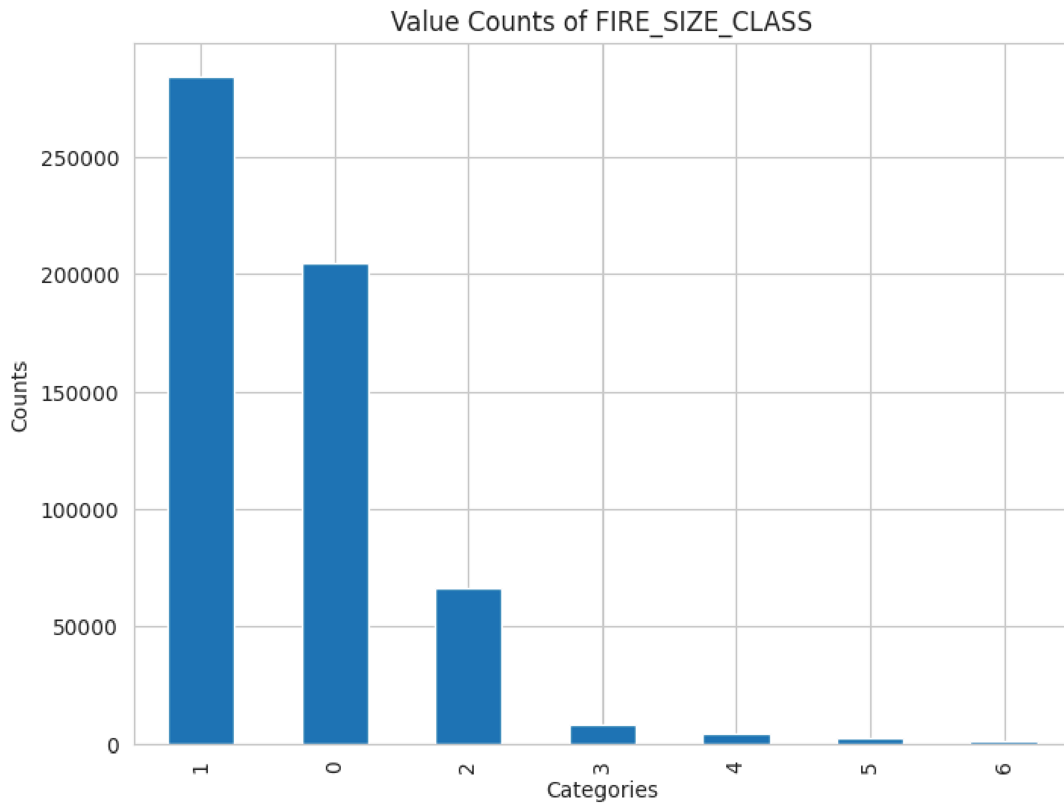
Competitive Lab in Data Science Final Project

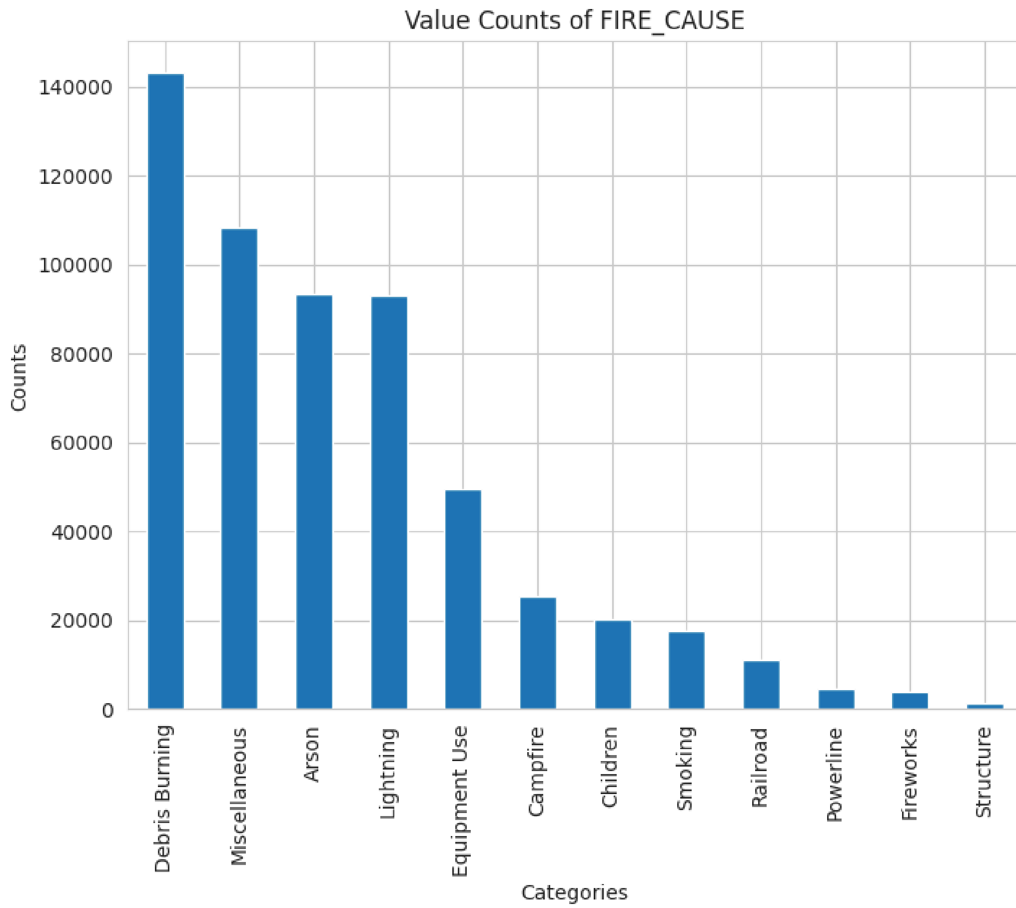
Karin Vashdi
Noam Tamam
Shahar Spencer

Initial Exploration & Data Visualization

First of all, in order to get a feel for the interaction between features and the target variable, we created different visualizations.
Below are some of them:

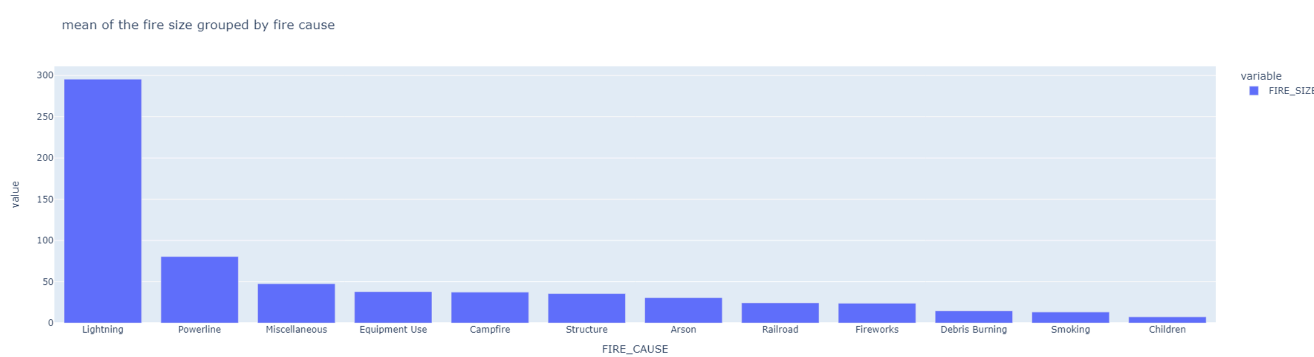


We can see from this visualization that there is an imbalance between the different classes of fire_size_class, there are many more smaller fires than large fires.



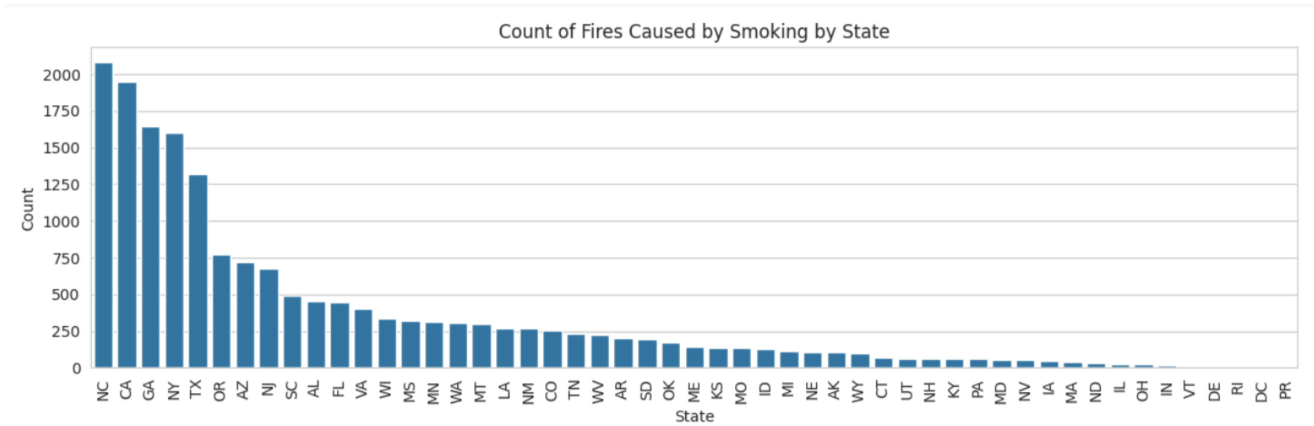
We can see from the graph above that there are some very common causes of fires, and some very rare causes of fires.

Another visualization we used to explore the data was the average fire size by the fire cause:



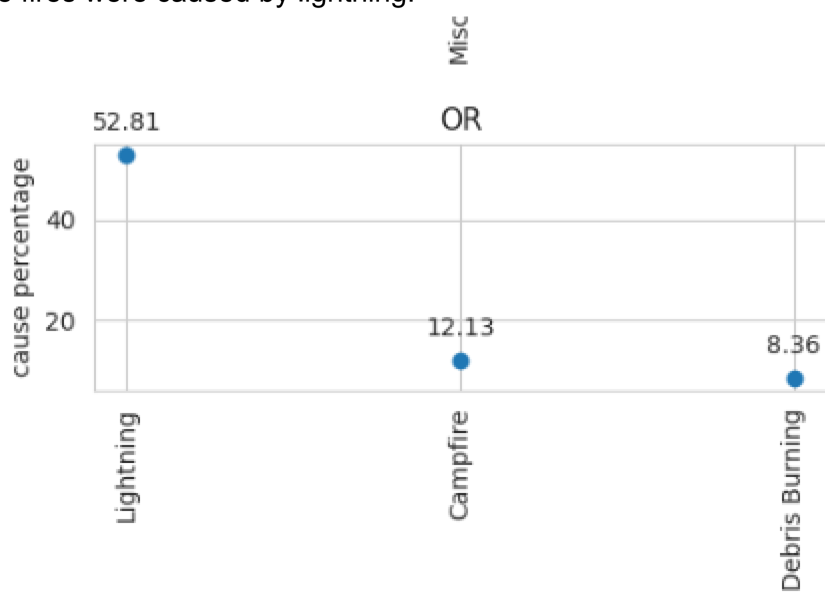
As we can see above, there are fire causes that cause very large fires on average (lightning), and fire causes that cause very small fires (such as children).

Additionally, we explored the relationship between fires caused by each one of the causes and the different states.
Smoking fire cause by state:



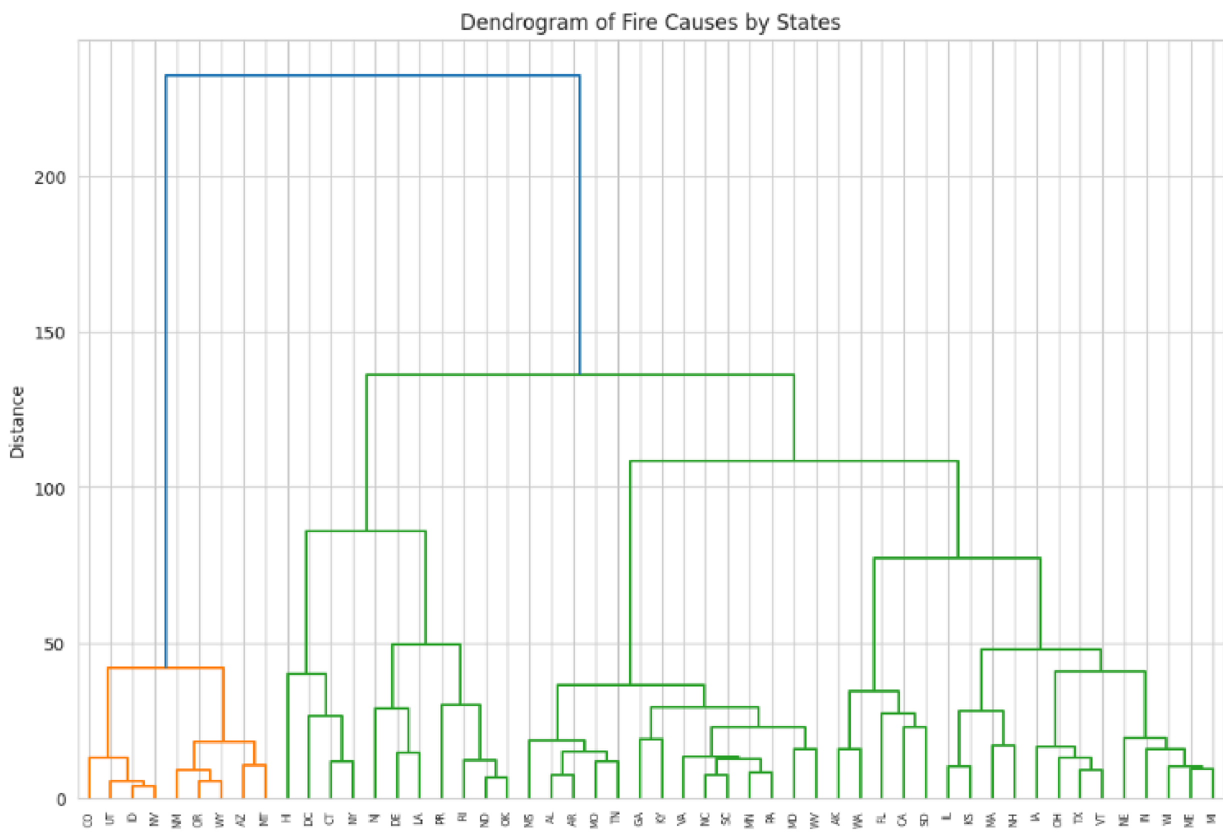
We created another set of graphs with the relative percentage that one fire cause caused a fire in that state. In order to make the graphs more readable we only took the top three.

An example graph is below. In this graph we see that in Oregon, 52.81% - over half - of the fires were caused by lightning:



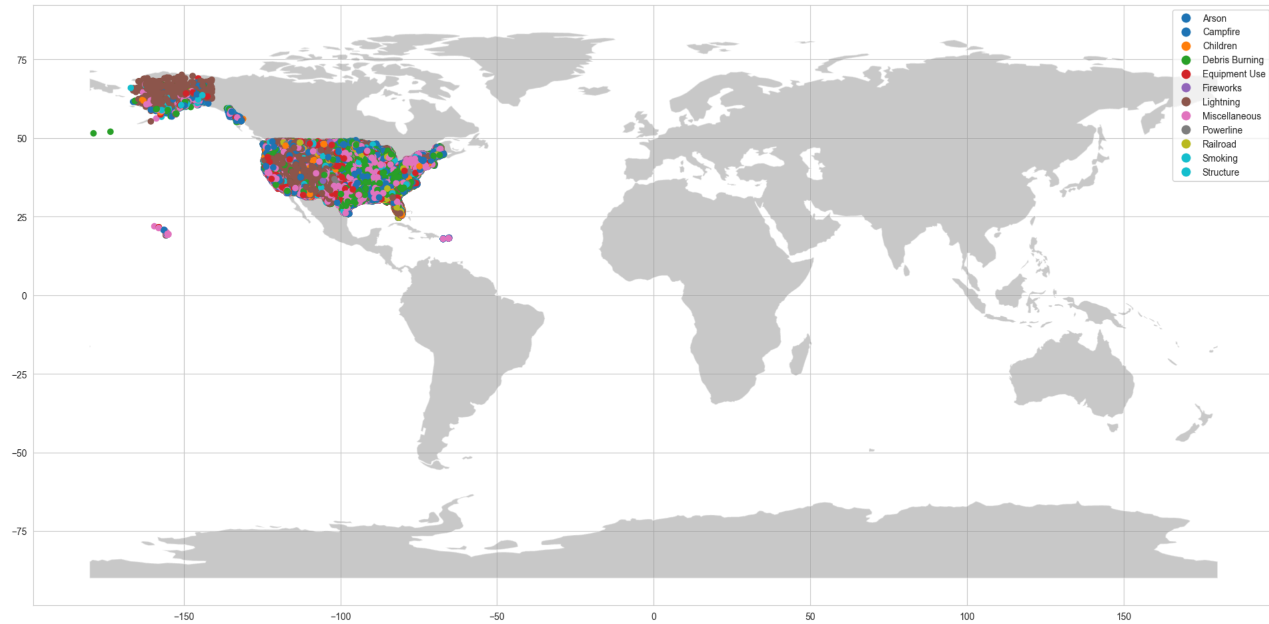
Additionally, we created a dendrogram depicting a clustering of the states, where each state is represented by a vector of the distribution of the fire causes (the vector adds up to 1, since it is a vector of probabilities).

The results were:



As we can see above, the states can be grouped quite nicely into clusters by the vector representing the distribution of the different fire causes.

Later on, we explored the spatial data:



Unfortunately, We couldn't conclude any new information from the map, due to technical challenges to integrate another layer to specific areas of the data.
If we could we would classify the areas as urban or forced areas.
Also, we tried to use K-means algorithm on the latitude and longitude features to group close fireplaces. The grouping didn’t contribute any meaningful information

Feature selection

We started off with deciding which features should be completely irrelevant for the prediction and removing them.
The rest of the features we would engineer, put into a model and explore the feature importance to see whether we should remove anything or engineer anything further.

Below is a table with all original columns in the dataset.
The “kept initially?” column reflects whether we deemed the column completely irrelevant, or decided to continue with it further.
The “feature type” column reflects the type of feature (numerical, categorical or else - in which case we must transform it into a type that can be fed to the model).
The “cleansed” column reflects whether we had to clean the column and if so how we did it.
The “possible engineering” column reflects possible features we can add based on this column (maybe combined with other columns as well).

Column name	**Kept initially?**	**Feature type**	**cleansed?**	**Possible engineering**
SOURCE_SYSTEM_TYPE	YES	categorical		
SOURCE_SYSTEM	YES	categorical		
NWCG_REPORTING_AGENCY	YES	categorical	No na values, no cleansing needed	
NWCG_REPORTING_UNIT_ID	NO, same info as NWCG_REPORTING_AGENCY			
NWCG_REPORTING_UNIT_NAME	NO, same info as NWCG_REPORTING_AGENCY			
SOURCE_REPORTING_UNIT	YES	categorical		

Column name	Kept initially?	Feature type	cleansed?	Possible engineering
SOURCE_REPORTING_UNIT_NAME	NO, same info as SOURCE_REPORTING_UNIT			
LOCAL_FIRE_REPORT_ID	NO, too specific			
LOCAL_INCIDENT_ID	NO, too specific			
FIRE_CODE	YES	Categorical		
FIRE_NAME	No, because it might cause leakage			
ICS_209_INCIDENT_NUMBER	NO, too specific			
ICS_209_NAME	NO			
MTBS_ID	NO			
MTBS_FIRE_NAME	NO			
COMPLEX_NAME	NO, could incur leakage from train to test			
FIRE_YEAR	YES	numerical	No, was clean	
DISCOVERY_DATE	YES	Date - need to convert to other	Yes, converted to usable date	Creating seasons of the year

Column name	Kept initially?	Feature type	cleansed?	Possible engineering
		format so can be fed to the model		Checking which holiday it was
DISCOVERY_DOY	NO		Reflects same info as discovery_date	
DISCOVERY_TIME	YES	numerical	Yes, put a default of -1 for null values	Might add time_of_day_category
STAT_CAUSE_DESCR	YES, converted name to fire_cause			
CONT_DATE	NO, almost half are empty, and the general information should come from the date of discovery			
CONT_DOY	NO, for the same reason			
CONT_TIME	YES	numerical		Might create feature of category of time of day
FIRE_SIZE	YES	numerical		
FIRE_SIZE_CLASS	YES	categorical		

Column name	Kept initially?	Feature type	cleansed?	Possible engineering
LATITUDE	YES	numerical	Yes	Combing latitude and longitude into coordinates
LONGITUDE	YES	numerical	Yes	Combing latitude and longitude into coordinates
OWNER_CODE	YES	categorical	No null values so no	
OWNER_DESCR	NO - same info as owner_code			
STATE	YES	categorical	No null values so none needed	
COUNTY	YES	categorical	About half were missing. We created a new category of “Unknown” for these.	
FIPS_CODE	YES	categorical	Many missing values which we mapped to the new category of “Unknown”.	
FIPS_NAME	NO, reflects same info as FIPS_CODE			

Baseline model

We decided to use gradient boosting. We chose the CatBoost algorithm which we learned about in class. We started with a baseline model with CatBoost's default parameters. This model was fed all features except the ones that were removed initially.

The score for this model was: 0.8528030571883328.

Feature engineering

We added the following features.

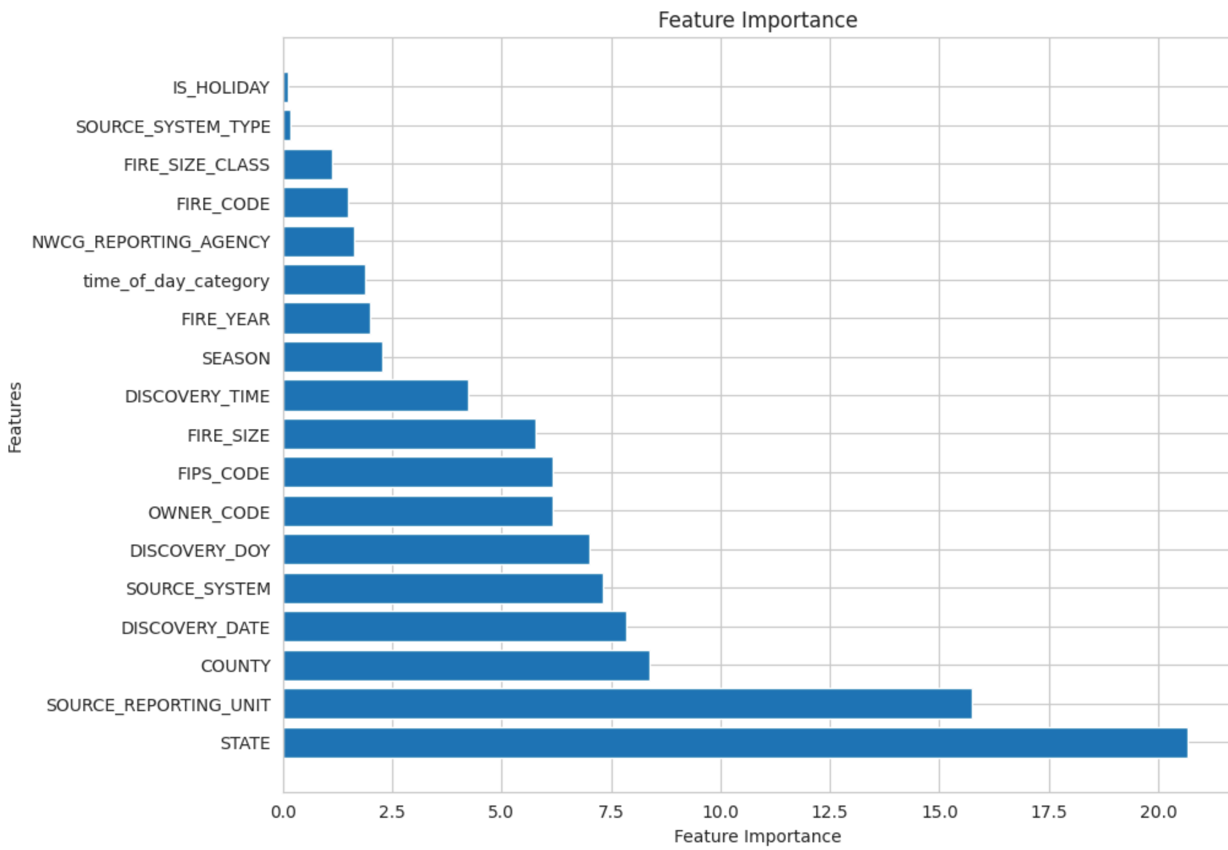
- Time of day category: categorize the time of the day that the fire was discovered into one of 7 categories, null values were mapped to a default "unknown" value.
- Season: was engineered from DISCOVERY_DOY into one of the four seasons. We might expect for example that fires that erupted in the summer might have a higher likelihood to be caused by campfires and fireworks, and fires caused by lightning occurred in the winter.
- Is_holiday: categorize whether or not the discovery date was a holiday. Again, this may help predict if fire was caused by fireworks, campfires and so on.

The score with the added features was: 0.8528232933020333, which is slightly higher than baseline.

Discovering feature importance

As we chose to provide features to the model with a minimum removal policy, we wanted to see whether removing features might benefit the model.

To do this, we reran the CatBoost model with the default parameters and all previous features, along with the engineered features, and got the feature importance from the baseline model:



It seems there are features with a near-zero benefit to the model, perhaps even hurting its performance.
However, there are features with very high importance, and with medium importance.

We took all features with an importance > 1.5, and tried all combinations of adding in the remaining features.
Initially we had a higher threshold > 3, and of the remaining features with lower importance we chose random subsets, but we saw that the model's performance went down, so we decided to lower the threshold.

we saw that the best score yielded was with the original features (along with the engineered ones): therefore we kept all features.

Hyper-parameter tuning

After exploring the data and the impact of the features, and choosing a beneficial subset of features, we began to tune CatBoost’s hyperparameters.
We used the optuna library, and also used cross-validation with k=5.
and tuned over many parameters available with catboost:

```
'iterations'
    'learning_rate'
    'depth'
    'l2_leaf_reg'
```

We couldn't find any improvement due to the parameters changing, the aocurcy result was even worse after the hyperparameters tuning

```
Best parameters found: {'depth': 8, 'iterations': 958,
'learning_rate': 0.003}
Best ROC AUC score found: 0.819137312659594
```

Therefore the best roc-curve accuracy score for the model is-
0.8528232933020333