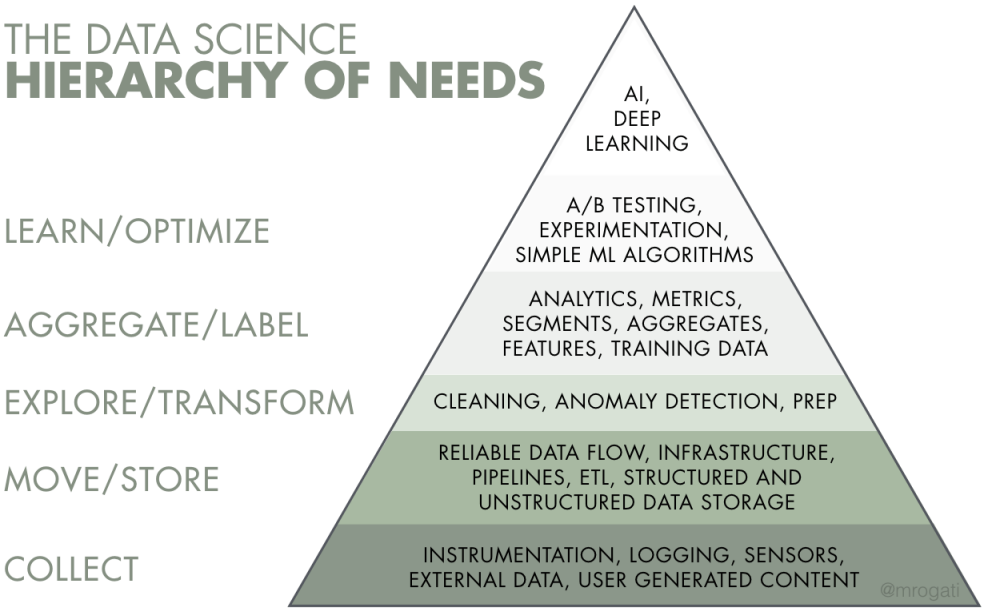
Data Science Project

Chaya M. Lasry 209911379
Eliav A. Dayanof 208674556
David Orenstein 209040112

February 24, 2023



Contents

1 Specifications

1. **Notebook** (ipynb)
2. **Report** (pdf)
3. **Requirements** (txt)

4. **Dependencies**
   (a) pure_functions.py
   (b) county_complete.csv
   (c) avg_precipitation.csv
   (d) avg_temp.csv
   (e) us_census_counties.shp (and its dependencies)

2 The Problem

Given a dataset of US wildfire records, predict the causes. Multi-class classification, the best classifier according to the weighted f1 metric wins.

TLDR – things we did:

- **Preprocessing**:
  - **Encoding**:
    - OHE – For us the natural choice for the 'STATE' feature was initially One-HotEncoding. But after empirical experimentation with it using tree based models, we noticed that oddly enough OHE reduces the model's score. After reading about it more in the literature we did see it wasn't a single occurnce and indeed tree based models **usually** perform worse with OHE on high cardinality features such as state.
    - Integer encoding
  - **Imputing**:
    - COUNTY: Using geo-spatial association with an external dataset
    - DISCOVERY_TIME: Using K-NN imputer
- **Feature Engineering**:
  - **Encoding**
  - **Extracting**
- **Algorithms**: Decided to go mainly for ensemble-tree based methods as they seem most effective for tabular data with numerical and categorical features.
  - **Random Forests - BEST PREFORMING**
  - **Gradient Boosting**
    - LightGBM (lightgbm as lgb)
    - XGBoost (xgboost as xgb, sklearn.ensemble.GradientBoostingClassifier)
- **Modeling techniques**:
  - Employed advanced imbalanced classification techniques such as:
    1. Synthetic Minority Oversampling Technique (SMOTE) – using imblearn (imbalanced-learn)
    2. Cost-Sensitive Learning for Multi-Class Classification – using random forest hyperparameter class_weight='balanc in sklearn.ensemble.RandomForestClassifier
  - Ensembling with stacking using sklearn.ensemble.StackingClassifier
- **Model and Feature Selection**:

  - Hyper-Parameter tuning: features, feature embeddings and feature representation was counted as a hyperparemeter in our holistic approach thus we optimized it alongside with the model hyperparameters using shap-hypetune
    * Grid search
    * Randomized search
    * Bayesian search

- **Libraries and specific advanced tools:**
  - **pandas**
    * groupby
  - **sklearn**
    * pipeline
  - **geopandas**
  - **shap-hyptune**
  - **imbalanced-learn**
    * SMOTE with a custom sampling_strategy=strategy parameter for oversampling the lesser classes compared to the frequent classes

3 Initial Data Analysis

In the initial data analysis stage we started familiarizing ourselves with the data by inspecting and visualizing it.

We divided our data analysis into a preliminary phase of *initial data analysis* and to the main analysis phase. The most important distinction between which is that during the initial data analysis we refrained from any analysis that is aimed at answering the original research question and were purely focused on assessing the **quality of the data**.

3.1 Data Specifications

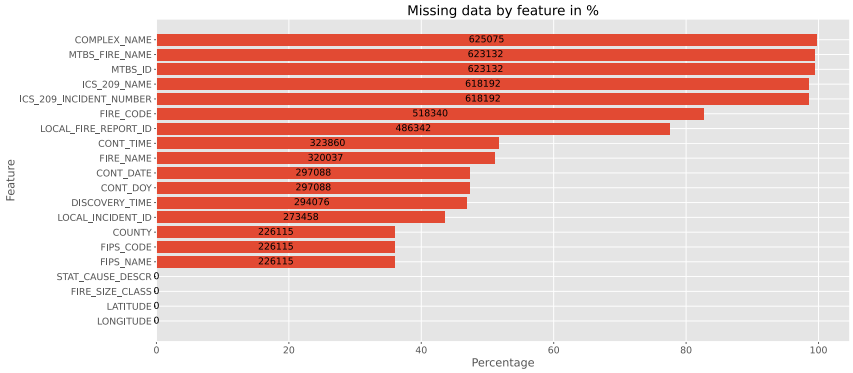
- What each record represents?
  - Wildfire incident.
- What is the target variable? Cause of fire represented by the following 13 classes:
  1. Lightning
  2. Equipment Use
  3. Smoking
  4. Campfire
  5. Debris Burning
  6. Railroad
  7. Arson
  8. Children
  9. Miscellaneous
  10. Fireworks
  11. Powerline
  12. Structure

13. Missing/Undefined

- How many variables do we have? around 40
- What kind of information do we have?
  - Geo-spatial related: State, latitude, longitude, county (along with identifiers), land owner info, etc...
  - Time related: Discovery date, discovery time, discovery day of year and also time data about when the fire was declared "contained".
  - Fire size
  - Unique Identifiers
- It's worth noting that the coordinates we are given are in the NAD83 datum, which means we are working on a specific coordinate system which is important when analyzing geo-spatial data between different databases.

3.2 Quality of data

- **Quality of data** – The quality of the data should be checked as early as possible. Data quality can be assessed in several ways, using different types of analysis: frequency counts, descriptive statistics (mean, standard deviation, median), normality (skewness, kurtosis, frequency histograms)
  - **Completeness**:



    Features worth noting

    1. County
    2. Discovery time
    3. Cont time
  - **Validity**: The degree to which the measures conform to defined business rules or constraints (see also Validity (statistics)).
    - lat and long: by the latitude and longitude plots we see they do align with the US borders ✓
    - County has numbers instead of county names sometimes

4 Data Preprocessing

Data, when initially obtained, must be **processed** or organized for analysis.

- **Encoding**:
  - OHE – For us the natural choice for the 'STATE' feature was initially One-HotEncoding. But after empirical experimentation with it using tree based models, we noticed that oddly enough OHE reduces the model's score. After reading about it more in the literature we did see it wasn't a single occurnce and indeed tree based models **usually** perform worse with OHE on high cardinality features such as state.

- Integer encoding

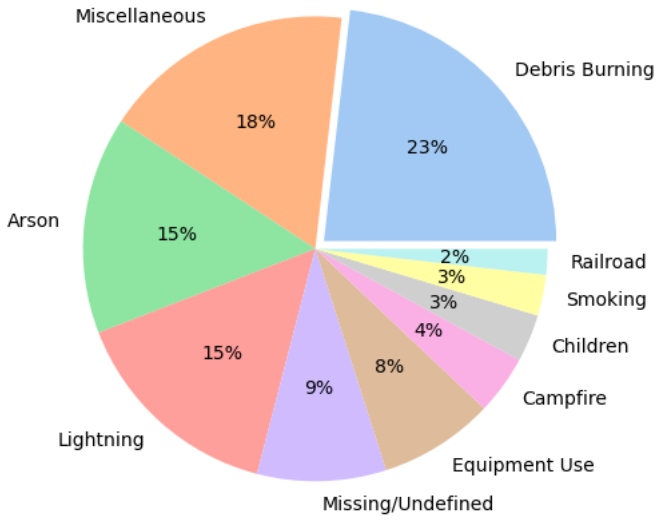
- Imputing missing values:
  - **County entries** – We used geo-spatial enabled dataframes using the geo-pandas library in order to impute the missing county entries using ground truth values we obtained from the US Census Bureau database. We did so by collecting a dataset of the shapes of the counties and by **spatially joining** the dataframes according to the points that were within the shapes.
  - **Discovery time** – we attempted to use k-NN imputer but it took a long time to train even for small k so instead we preformed iterative imputer.

5 Data Exploration (EDA)

In statistics, **exploratory data analysis** (EDA) is an approach of analyzing data sets to summarize their main characteristics, often using statistical graphics and other data visualization methods. This is the main analysis phase, analyses aimed at answering the research question are performed as well as any other relevant analysis needed to write the first draft of the research report.

5.1 Visualizations

Label Distribution



1. Over 20%: Debris Burning
2. 10%-20%: Miscellaneous, Arson, Lightning
3. Under 10%: Missing/Undefined Equipment Use, Campfire, Children, Smoking, Railroad
4. under 2%: Fireworks, Powerline, Structure

(more on the notebook)

Visualization Conclusions

1. Correlations:
   (a) Correlation of day of weak with label
      i. Hypothesis: Weekends are highly correlated to the campfire class
   (b) Correlation of day of year with label
      i. Hypothesis: 4th of july is highly correlated with fireworks
2. Feature to label graphs:
   (a) Distribution of labels per day of year, can be done s.t each bar in the histogram is a pie chart with the distribution of label per the particular day

6 Data Engineering

Feature engineering or **feature extraction** or **feature discovery** is the process of using domain knowledge to extract features (characteristics, properties, attributes) from raw data. This stage is done after EDA, and uses the insights gained from it.

Besides basic transformations that we count as preprocessing nothing in our **own** data seemed extremely interesting so we turned to external data sources and to **data collection**.

Because most causes of fire are a product of human related activities, domain knowledge will indicate that socioeconomic, demographic and geo-social related factors are associated with reckless behavior, crime rates, high birth rate and smoking rate which are quite likely to be associated to their corresponding causes of fire, i.e debris burning, arson, children and smoking respectively.

Data Collection: We collected a variety of datasets from many domains, based on equired domain knowledge and research on the task at the hand, which turned out to be much more challenging than we thought, these include:

- Time related
  - US Holidays
  - **Special events - (big sporting events, concerts)**
- Geo-spatial
  - Elevation
  - Climate
  - Vegetation
  - Land-cover
- Geo-meteorological and time related
  - Temperature
  - Precipitation
  - Air Pressure
  - Moisture
  - Wind
- Geo-demographic and Geo-socioeconomic
  - Income

- Children rate
- Crime rates
- Smoking statistics

Unfortunately those marked at red were proven too difficult at this time, because of technical reasons, data availability and time constraints.

The data was obtained from many sources, including:

1. National Wildfire Coordinating Group (NWCG)
2. United States Census Bureau
3. National Oceanic and Atmospheric Administration (NOAA)
4. National Centers for Environmental Information (NCEI)

Our workflow to gather was as follows:

1. Imputing the counties using geo-spatial techniques
2. Collecting geo-spatial data, or any other data by US counties
3. Merging the databases

7 Models and Algorithms

7.1 Theory

1. Data Exploration
2. Data Preprocessing
3. Feature engineering
   (a) Feature Extraction
   (b) One Hot Encoding
   (c) Feature Scaling
   (d) Feature Learning
   (e) Feature Imputation
4. Baseline models
   (a) Random Forest Classifier
   (b) Logistic Regression
   (c) Support Vector Machine

8 What could we have done better?

8.1 Observations, Ideas, Remarks, Realizations and Hypothesis (hafirot)

1. Heuristic for ML:
   (a) Very rough feature engineering to make the model learnable
   (b) Baseline:
      i. Constant classifier based on most common class
      ii. Random classifier based on label distribution
   (c) Data Preprocessing
      i. Data Classification -
         A. classify the features based on their type (Categorical, Numerical, Ordinal, Unique),
         B. get a genearl understanding of what each of them means
         C. what format
         D. specific type
      ii. Data Visualization - understanding trends, corrolations, feature importnece
         A. Trendes compared to the label for good features
      iii. Data Cleaning and fillna'ing
      iv. Feature Engineering
         A. Feature Extraction
         B. From existing data
            - adding is_holiday feature
            - adding is_summar
            - adding is_winter
         C. From outside sources
            - adding socioeconomic_rank by region
            - adding height feature
         D. Dropping Features
   (d) Machine learning
      i. Creating a baseline model and getting a base evaluation of something simple
      ii. Choosing a model by:
         A. Training different model
         B. Evaluateing them
         C. Comparing the weighted F1 metric
      iii. Hyper-Parameter tuning

- When looking at the fire causes it's important to distinguish between **natural causes** and **human causes** maybe we can even try to classify natural vs human first
  - human_accessibility feature based on location, could be based on
    * distance_to_roads
- **Location** and **time** probably give us the most predictive power on this dataset.
- The labels, i.e. t**he causes of fire are not rigorously defined**, it is not entirely clear what some of the causes mean, for example: Equipment Use, Structure, Railroad, Miscellaneous. Possible courses of action:

- Perhaps we should research the original website or the web in general for farther clarifications on the exact definitions - found good shit, added it to "domain_knowledge" folder

- When considering adding a certain feature it is very useful to ask and answer the following questions:
  - **Hypothesis**: What is a sensible hypothesis from which we conclude adding this feature might be worth it?
    This hypothesis is best made using **"Domain Knowledge**" but when domain knowledge is lacking it could also be made by making a "common sense assumption" which should be justified.
  - **Possible Justifications**: How can we justify the hypothesis or its assumptions?
    - **Statistics** which could be presented by
      - **Visualizations**
      - **Statistical Analysis**
    - **Domain knowledge**
    - **Evaluations** - adding the feature and checking if it improved performance, if it did and our hypothesis was that this feature would be a good predictor variable then we have a plausible affirmation (never a definite confirmation). But if our hypothesis was something else related that if was true would improve performance then we have no way to know if it's because our hypothesized thing or a different thing.
  - **Possible Implementations**: How should we implement adding this feature?
    If it's a categorical value then maybe use onehot encoding, if we need to use external data then how should we do it most effectively and so on.

- Possible ideas for the data engineering part which could help the prediction of a specific label:

1. Lightning - Natural
   (a) Weather (location and time based): either general quantitative weather stats or qualitative like:
   - **season** –
     - **hypothesis**: sensible hypothesis is that lighting occur more often in the winter
     - **possible justifications:** bar plot of lighting count per seasons, meteorological domain knowledge
     - **possible implementations:** possibly a onehot variable encoding the 4 main seasons (summer, winter, fall, spring)
   - **close_to_big_tornado** (location and time based) -
     - **hypothesis:** sensible hypothesis is that lighting occur more often when a tornado is close (both geographically and in time)
     - **possible implementation:**
       - source a list of tornadoes
       - sort them by overall intensity using accepted metrics (and perhaps filter only those who pass a certain intensity threshold)
       - extract day by day breakdown of intensity by location
       - define a proximity metric which weighs in both location and time proximity somehow
     - **possible justifications:** meteorological domain knowledge

   (b) **time_of_day**
   i. hypothesis: during the night lighting occurs more often
   ii. possible justifications: meteorological domain knowledge, correlation plots
   iii. possible implementation: already exists as "DISCOVERY_TIME" but only in about 55% of the records, could very possibly **impute** the rest (possible approaches detailed below)

2. Equipment Use - Human
3. Smoking - Human
    (a) smoke_rate - using smoking statistics per state (or even by county?)
4. Campfire - Human
5. Debris Burning
6. Railroad - Human
    (a) railroads_acres - try to look up the amount of acres squared are ocuppied by railroads or maybe define some other metric like: the proportion or log proportion (to scale, or maybe use standerd_scaler instead) of railroad acres out of total country size in acres
7. Arson - Human
8. Children - Human
9. Miscellaneous
10. Fireworks - Human
    (a) possibly indicative features:
        i. close_to_independence_day
11. Powerline
12. Structure
13. Missing/Undefined

- **Location:**
  - State stats such as:
    * State size in acres
- **Time:**
  - **day_of_year** is quite possibly a good predictor because of reoccurring characteristics of the year:
    * Holidays - human causes
    * Seasons - natural causes
- **Time and Location:**
  - Special Events (that reoccur at a certain date each year)
  - **Weather**:
    * **Temperature** – Irregular temperature, very hot, very cold could in general be a good indicator of the likliness of fire, not necessarily of a particular cause
    * **Precipitation** – Rainy, snowy, cloudy, clear, wet, dry, calm
    * **Air pressure**
    * **Moisture**
    * **Wind**
  - Day

        * Special events - natural/human causes
            · Earthquakes
            · Tornados - big torandos that occured
            · Big sport events
            · Terror attacks

- **Important to remember:** If we visualize number of fires per state for example we need to normalize for things, like size of the state and the population count
- **Wild ideas:**
    - create a model for every state and then create an ensemble that will weigh in all the models for the final result
    - Classify in parts:
        * perhaps first classify by natural causes or human causes, then in human, accidental or malicious, then by class
- **Possible leakage-incurring features**: According to Schifter we aim on predicting the cause of fire preferably when it starts. Some When it's not clear yet, or possibly also after it, but we need to this into consideration and show that we did. Perheps we can make 3 models, 1 for each phase of the fire, and gradually include more features that will be known at a later time after than when the fire started.
    - **Final fire size** related features
        * FIRE_SIZE
        * FIRE_SIZE_CLASS
    - **Final fire time** related features of the form CONT_*, cont=contained
        * CONT_DATE
        * CONT_DOY - contained day of year
        * CONT_TIME

    It is important to note however, we can still extract features from this data which wouldn't count as leakage such as:
    - average fire size by state (in acres or in class)
    - average fire length by state (in time)

    which if we assume that the test data doesn't deviate that much then it's another state attribute that could be a good indicator assuming some causes are associated with larger or longer lasting fires.
- Almost half of the records have missing DISCOVERY_TIME, it is very possible to **impute wisely** using statistical methods or k-NN that predicts the time. Naive methods:
    - Simple statistical methods: impute the mean, median, mode per state or something
    - Advanced statistical methods: we don't know yet, need to research.
        * General idea: check the empirical distribution of the the discovery times over the dataset we know and infer the rest using it in some fashion if it seems representative.
    - Additional ML model: use another ML model to predict the time

- Easily dropped features: Features that seem very insignificant or even harmful for the learning process ("garbage in garbage out"), characteristics

  1. They are **unique identifiers** or **unique names** which without deeper analysis seem entirely useless in a ML settings.
  2. A lot of **missing values**

  specifically the following features follow these characteristics and have close to 100% missing values therefore are definitely going to be dropped: COMPLEX_NAME, MTBS_FIRE_NAME, MTBS_ID, ICS_209_NAME, ICS_209_INCIDENT_NUMBER

- Visualizations to add:
  - hbar plots:
    - Number of fires by cause (in thousands)
    - Average fire size by cause (in acres)
    - Average fire duration by cause (in days)
  - Heat maps:
    - Number of fires by state (so we know to normalize by this number) - hypothesis: probably California and Texas will have the most fires due to size and climate.
      - average number of fires in a year by state
    - Average fire size by state (in acres)
    - Average fire duration by state (in days)

- **Owner code** can be used to determine natural vs human likeliness.

- Initial Data analysis and preprocessing:
  - Error checks
    - visual inspection of the mapped lat and long points on the map to see if it is reasonable and no mistakes are made
    - flag lat and longs which mismatch the proclaimed state territory as potential droppers
    - clearly erroneous dates
    - O or I turning to 0 or 1 or vice versa
  - Redundancy

- Worth checking as a proof of concept: what happens if we use the most frequent dummy classifier on each state and then ensemble the result at the end.

- Lighting fires' duration is much longer than other causes. Because they usually occur in very remote locations, which are not easily accessible by humans, therefore the fire goes undetected for long until someone notices it.

- Outliers analysis, after we've done the bulk of the work we should do an analysis on hard to predict examples, explain them,
  - SHAP
  - Gini impurity

  - Partial Dependence Plots(PDP)
- States - OHE worked worse, counter intutitive possible reasons:
  - Curse of dimensionality - more dimensions make it harder for the model, a trade-off between signal and dimensions
  - The worse performance is more representative of the truth, rather than pure luck. could be tested using:
    * Cross validation
    * different randomization of the state variable
- Experiment with imbalanced sklearn and other shit that was presented in week7 lectures.
- Time-series analysis
  - Check what happens if we leave the date in the Julian format
- Geo-spatial analysis
  - Topography - data
    * maps of roads, raliroads
  - Meteorlogy
- # DS_HP is our keyword for every feature representation HP
- We need to check for duplicates
- Natural
  - Lighting
- Human
  - Intentional
    * Arson
  - Accidental
    * Debris Burning
    * Fireworks
    * ...

1. Features:
   (a) Geo-spatial
      i. elevation
      ii. vegetation
      iii. landcover
   (b) Time
   (c) Geo-spatial and time
      i. Weather
         A. **Temperature** – Irregular temperature, very hot, very cold could in general be a good indicator of the likliness of fire, not necessarily of a particular cause
         B. Precipitation – Rainy, snowy, cloudy, clear, wet, dry, calm
         C. **Air pressure**
         D. **Moisture**
         E. **Wind**

9 Workflows

During our work throughout the course we gained quite a bit of hands-on experience with data which enabled us to formulate a few workflows we found were very effective for us during this project which we would love to share.

We think these workflows sum up well what we learned and based on the tools we currently posses propose a very effective framework of tackling any machine learning oriented data science project.

9.1 General Workflow

1. Perform **data preparation**
2. Continue with getting a **baseline** model running
3. Iteratively:
   (a) Perform the **data science** workflow
   (b) Perform the **machine learning** workflow
4. **Evaluate** the best performing candidates over all the possible data and model combinations on the test set (which was never touched before at any earlier stage)
5. Out of these **pick the best** performing one

9.2 Data Preparation Workflow

Understanding the data at the shallow level first. Understanding the data at this level may not be data science yet but it is essential in order to do any data science.

1. **Data Collection** (optional)
2. **Initial Data Analysis (IDA)**
3. **Data Preprocessing**
   (a) data-cleansing
   (b) split the dataset

9.3 Baseline Workflow

Also known as the **minimal working example** workflow aimed at getting a **baseline** model that is simply aimed at successfully returning an output based on the input data we give it, a simple **sanity check**, convince ourselves that we can at the very least run the ML algorithms we wish to extensively compare between on the given data as quickly as possible.

1. **Feature Dropping** – Drop all but the absolute essential features that you we are definitely not going to drop.
2. **Basic Data Preparation** – Simply make sure that the data that is left is capable of being used as an input to a ML algorithm without errors, final format corrections if needed.
3. **Model Testing Framework** – Prepare the code for running all the popular models (classifiers at our case) such as: Dummy Classifier, XGBoost, Random Forest, SVM, Decision Trees, Naive Bayes, Logistic Regression, etc. With factory reset configurations and hyperparameters!

4. **Evaluation Framework** – Run all the models on the data and prepare an evaluation framework. Make sure that:

   (a) Training and prediction occur with no errors.

   (b) Evaluation is done with a validation set or with CV (training phase eval).

   (c) We are getting reasonable results assuming the data is very minimal and the models are factory reset

9.4 Evaluation Workflow

Evaluation should occur in almost all stages. But only one time, at the very end, we simply evaluate on the **test set** and pick the best model. At any preceding phase we need to evaluate using a different methodology described below. Reason being we wish to avoid "contaminating" the process with bias that could very easily lead to **overfitting**.

So weather it's feature selection, feature engineering, hyperparameter-tuning, regularization, algorithm selection or any other phase which involves making decisions which are based on model attributes, we completely avoid touching the test set or even seeing it and instead evaluate using either of the following approaches:

- **Validation Set** – A subset of the dataset reserved purely for hyperparameter-tuning.

  1. Training is done on the train sets
  2. Testing is done on the val sets

- **Cross Validation** – K-fold cross validation or other CV techniques we don't know yet.

  1. Reserve a test set (no val set needed)
  2. Partition the data to k folds, train on $k-1$ and test on the 1 left out k times, average the result

9.5 Data Science Workflow

In this workflow we **disregard the algorithm selection** phase and focus on answering the following **data oriented questions**: which features to drop? which features to add? what is the optimal feature representation? should more data be added from external sources?

We perform the stages of this workflow assuming the following assumptions:

1. We are **given a configured model** (classifier) with a specific pre chosen set of hyperparameters.
2. We are **not going to change the classifier at all**, obviously not to a different classifier but also not by tweaking a single hyperparameter.

Following these assumptions will enable us to control for the variable at hand which at this point is only the **data**.

- **EDA** – Begin understanding the messages contained within the obtained data.

  - Visualize the connection between holidays and different labels
  - Identify visually interesting things you see and hypothesize hypothesis based on it
  - Maybe statistical analysis combined with domain knowledge that will help solidify the hypothesis

- **Feature engineering**

1. Raise data engineering ideas that will use the conclusions from the EDA to create possibly indicative features, for example:
   (a) feature engineering
   (b) adding features
   (c) etc.
2. If external **data collection** is needed to implement the ideas perform the data preparation workflow (**iterative process**)
3. Implement the feature engineering

- **Evaluating the changes** – the idea is to think about **features as hyperparameter** and about the evaluation of their predictive power as **hyperparameter-tuning**.
  - Evaluate each change that was made by:
    * Gradually adding features
    * Comparing each change to the past
    * Trying different combinations of changes like we do in **grid search**

Example:

1. Hypothesis: holidays could be very predictive of certain labels:
   (a) EDA: which showcases correlations between proximity to certain holidays with specific causes of fire, for example 4th of july and fireworks
   (b) Feature engineering:
      i. collecting holiday data
      ii. creating a "is_close_to_holiday" feature for holidays which seemed like good predictors
   (c) Evaluating the performance after the feature engineering and comparing it to the performance before

9.6 Machine Learning Workflow

Algorithm selection and model hyperparameter-tuning. We assume we have data and features that are not going to change, and we perform hyper parameter-tuning or possible algorithmic adaptations or variations to test the best algorithm suited for the job.

1. Perform algorithm changes:
   (a) Hyperparameter-tuning using either of the following:
      i. Grid search
      ii. Randomized search
      iii. Bayesian Search
   (b) Regularization
   (c) Personal variation on the algorithm itself (probably not)
2. Evaluate using the evaluation workflow

10 Appendix.

10.1 The Dataset

- This data publication contains a spatial database of wildfires that occurred in the United States from 1992 to 2015.
- It is the third update of a publication originally generated to support the national Fire Program Analysis (FPA) system.
- The wildfire records were acquired from the reporting systems of federal, state, and local fire organizations.
- The following core data elements were required for records to be included in this data publication:
  - discovery date
  - final fire size
  - point location
    - With precision at least as precise as Public Land Survey System (PLSS) section (1-square mile grid).
- The data were transformed to conform, when possible, to the data standards of the National Wildfire Coordinating Group (NWCG).
  - Basic error-checking was performed and redundant records were identified and removed, to the degree possible.
  - The resulting product, referred to as the Fire Program Analysis fire-occurrence database (FPA FOD), includes 1.88 million geo-referenced wildfire records, representing a total of 140 million acres burned during the 24-year period.

10.2 Features

1. STATCAUSECODE = Code for the (statistical) cause of the fire.
   (a) STATCAUSEDESCR = Description of the (statistical) cause of the fire.
2. Time related:
   (a) • FIRE YEAR = Calendar year in which the fire was discovered or con- firmed to exist.
   (b) • DISCOVERY DATE = Date on which the fire was discovered or confirmed to exist.
   (c) • DISCOVERY DOY = Day of year on which the fire was discovered or confirmed to exist.
   (d) • DISCOVERY TIME = Time of day that the fire was discovered or con- firmed to exist.
      i. 4th of july correlated probably to fireworks
   (e) • CONT DATE = Date on which the fire was declared contained or other- wise controlled (mm/dd/yyyy where mm=month, dd=day, and yyyy=year).\
      i. we can extract the length of the fire period
   (f) • CONT DOY = Day of year on which the fire was declared contained or otherwise controlled.
      i. numerical 1-365

ii. summar and hot days is a good predictor for cretain causes

iii. winter could be correlated to lighting

(g) • CONT TIME = Time of day that the fire was declared contained or oth- erwise controlled (hhmm where hh=hour, mm=minutes).

3. Geographical

(a) • GeographicArea = Two-letter code for the geographic area in which the unit is located (NA=National, IN=International, AK=Alaska, CA=California, EA=Eastern Area, GB=Great Basin, NR=Northern Rockies, NW=Northwest, RM=Rocky Mountain, SA=Southern Area, and SW=Southwest). •

(b) Gacc = Seven or eight-letter code for the Geographic Area Coordination Center in which the unit is located or primarily affiliated with. •

i. categorical represnted as codes for geographic locations

(c) LATITUDE = Latitude (NAD83) for point location of the fire (decimal degrees).

(d) LONGITUDE = Longitude (NAD83) for point location of the fire (decimal degrees).

i. Latitude, Longitude, and Temperature

(e) Country = Country in which the unit is located (e.g. US = United States). •

i. geographical

(f) State = Two-letter code for the state in which the unit is located (or primarily affiliated). •

i. geographical

(g) • STATE = Two-letter alphabetic code for the state in which the fire burned (or originated), based on the nominal designation in the fire report.

(h) • COUNTY = County, or equivalent, in which the fire burned (or origi- nated), based on nominal designation in the fire report.

4. Fire Characteristics:

(a) FIRE SIZE = Estimate of acres within the final perimeter of the fire. •

(b) FIRESIZECLASS = Code for fire size based on the number of acres within the final fire perimeter expenditures (A=greater than 0 but less than or equal to 0.25 acres, B=0.26-9.9 acres, C=10.0-99.9 acres, D=100-299 acres, E=300 to 999 acres, F=1000 to 4999 acres, and G=5000+ acres).

5. Unique Identifier

(a) FOD ID = Global unique identifier.

i. Unique

(b) FPA ID = Unique identifier that contains information necessary to track back to the original record in the source dataset. • SOURCESYSTEMTYPE = Type of source database or system that the record was drawn from (federal, nonfederal, or interagency). •

i. Unique, may be helpful to extract more info from the source data

6. Service Handlers/Governmenetal

(a) • FIPS NAME = County name from the FIPS publication 6-4 for represen- tation of counties and equivalent entities.

(b) • NWCGUnitIDActive20170109: Look-up table containing all NWCG iden- tifiers for agency units that were active (i.e., valid) as of 9 January 2017, when the list was downloaded from https://www.nifc.blm.gov/unit id/Publish.html and used as the source of values available to populate the following fieldsin the Fires table: NWCGREPORTINGAGENCY, NWCGREPORTIN- GUNITID, and NWCGREPORTINGU- NITNAME.

(c) • UnitId = NWCG Unit ID.

(d) WildlandRole = Role of the unit within the wildland fire community. •

(e) UnitType = Type of unit (e.g., federal, state, local). •

(f) Department = Department (or state/territory) to which the unit belongs. •

(g) Agency = Agency or bureau to which the unit belongs. •

(h) Parent = Agency subgroup to which the unit belongs (A concatenation of State and Unit from this report - https://www.nifc.blm.gov/unit id/publish/UnitIdReport.rtf). •

(i) Code = Unit code (follows state code to create UnitId). •

(j) Name = Unit name.

(k) • FIPS CODE = Three-digit code from the Federal Information Process

7. Names of stuff:

(a) SOURCESYSTEM = Name of or other identifier for source database or system that the record was drawn from. See Table 1 in Short (2014), or .pdf, for a list of sources and their identifier.

i. Categorical, may be helpful

(b) SOURCEREPORTINGUNIT NAME = Name of reporting agency unit preparing the fire report, based on code/name in the source dataset.

(c) FIRE NAME = Name of the incident, from the fire report (primary) or ICS-209 report (secondary).

(d) MTBSFIRENAME = Name of the incident, from the MTBS perimeter dataset. •

(e) COMPLEX NAME = Name of the complex under which the fire was ultimately managed, when dis- cernible.

(f) OWNER DESCR = Name of primary owner or entity responsible for man- aging the land at the point of origin of the fire at the time of the incident. Standards (FIPS) publication 6-4 for representation of counties and equiv- alent entities.

8. ID's

(a) NWCGREPORTINGUNIT ID = Active NWCG Unit Identifier for the unit preparing the fire report. •

i. need to check further

(b) LOCALFIREREPORT ID = Number or code that uniquely identifies an incident report for a particular reporting unit and a particular calendar year.

i. need to check further

(c) • LOCALINCIDENTID = Number or code that uniquely identifies an inci- dent for a particular local fire management organization within a partic- ular calendar year.

i. need to check further

ii. so far everything was about who reported the thing

(d) MTBS ID = Incident identifier, from the MTBS perimeter dataset. •

9. Codes:

(a) • OWNER CODE = Code for primary owner or entity responsible for man- aging the land at the point of origin of the fire at the time of the incident.

(b) • FIRE CODE = Code used within the interagency wildland fire commu- nity to track and compile cost information for emergency fire suppression (https://www.firecode.gov/). • 4

i. need to check further

10. Misc.

(a) NWCGREPORTINGAGENCY = Active National Wildlife Coordinating Group (NWCG) Unit Identifier for the agency preparing the fire report (BIA = Bureau of Indian Affairs, BLM = Bureau of Land Management,

i. Categorical, doubtful importence

(b) BOR = Bureau of Reclamation, DOD = Department of Defense, DOE = Department of Energy, FS = Forest Service, FWS = Fish and Wildlife Service, IA = Interagency Organization, NPS = National Park Service, ST/C&L = State, County, or Local Organization, and TRIBE = Tribal Organization). •

i. Categorical, need to check further

(c) NWCGREPORTINGUNIT NAME = Active NWCG Unit Name for the unit preparing the fire report. SOURCEREPORTINGUNIT = Code for the agency unit preparing the fire report, based on code/name in the source dataset.

i. need to check further

(d) ICS209INCIDENT NUMBER = Incident (event) identifier, from the ICS- 209 report.

(e) ICS209NAME = Name of the incident, from the ICS-209 report. •

10.3 Resources

- https://www.nwcg.gov/data-standards/approved
- https://www.nifc.gov/fire-information/fire-prevention-education-mitigation/wildfire-investigation
- https://en.wikipedia.org/wiki/Bureau_of_Land_Management - BLM
- BLM and lighting geo locations are highly correlated, makes sense because BLM areas are mostly
- https://developers.arcgis.com/python/samples/historical-wildfire-analysis/
- https://towardsdatascience.com/train-validation-and-test-sets-72cb40cba9e7
- https://towardsdatascience.com/shap-for-feature-selection-and-hyperparameter-tuning-a330ec0ea104
- https://stats.stackexchange.com/questions/264533/how-should-feature-selection-and-hyperparameter-optimization-be-ordered-in-the-m
- SHAP-hypetune – a library for robust hyper-parameter tuning and feature selection: https://github.com/cerlymarco/shap-hypetune

- Ways to encode categorical features – https://towardsdatascience.com/6-ways-to-encode-features-for-machine-learning-algorithms-21593f6238b0
- Leakage – https://towardsdatascience.com/data-leakage-in-machine-learning-how-it-can-be-detected-and-minimize-the-risk-8ef4e3a97562
  - Leakage related to time?? What if we want to predict the cause of fire on an old fire, our model has information from the future also.
    - well this information is actually even not necessarily entirely relevant and could actually hinder the performance so it doesn't incur leakage.
- PyGIS - Open Source Spatial Programming & Remote Sensing textbook: https://pygis.io/docs/a_intro.html
  - for creating an online textbook documentation docs style similar to this:
    - https://jupyterbook.org/en/stable/intro.html
    - https://github.com/executablebooks/cookiecutter-jupyter-book
- https://plotly.com/python/plotly-express/
  - https://dash.plotly.com/
- Imbalanced classification techniques –
  - https://machinelearningmastery.com/multi-class-imbalanced-classification/
    - SMOTE - Synthetic Minority Oversampling Technique: https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/
      - I've experimented with this and it seemed to not work as well
    - Cost-Sensitive someting something, class_weight HP
  - Multiclass imbalanced learning with one-versus-one decomposition and spectral clustering – https://www.sciencedirect
- OHE on categorical features in trees is bad in practicallity – https://notebook.community/roaminsight/roamresearch/BlogP
- Stacking Ensemble – https://machinelearningmastery.com/essence-of-stacking-ensembles-for-machine-learning/