Drive To Survive

An Analysis of Vehicle Failure in Formula 1 Auto-Racing

Contributors:
Monica Williams, Georgetown University
James Bifulco, Georgetown University
Bill Corkery, Georgetown University
Thomas Marzol, Georgetown University

Project Github Link

(Morlidge, 2020)

Contents

Throughout this paper, callout boxes on the right like this one will appear. Grey callouts will provide additional commentary on subjects being discussed to the left

Red callouts boxes will denote key ideas

Blue callouts will highlight underlying assumptions that the analysis is dependent on

Abstract

Our team built several classification models, of which Bagging Classifier was the best, to predict whether a car in Formula 1 would finish the race or not. We gathered a range of data on race results from 1996 to the present day, including features such as car speed, weather, and track features. Our data underwent a variety of transformations and wrangling until we were able to implement the most optimized versions into our models, which we measured with ROC curves and confusion matrices. Our analysis identified patterns in vehicle failure and uses these to make predictions.

Our primary goal was to predict whether a car in Formula 1 would finish the race or not

Executive Summary

Formula 1 is an international auto-racing circuit composed of single-seater cars that are constructed based on a "formula" of rules organized and managed by sport's governing body. The sport is currently composed of 10 teams, known as Constructors, with two Drivers and two cars per team in each race. Beyond these core features, most of the rules, circuits, Driver lineups, team structures, and financial systems of the sport are constantly in flux. As a result, the sport has been resistant to the implementation of data science. Our team sought to help change that.

Formula 1 does not have a level financial playing field. Each team has an independently sized budget that it draws from to build its cars and pay its Drivers. As a result, every vehicle failure matters significantly to teams both from a short-term perspective of on-track competition and the long-term perspective of financial stability and strength. Our team's objective was to understand the factors that influenced the probability of a Formula 1 car completing a given race successfully without experiencing a collision, accident, mechanical failure, or any other race-ending failure. We built a dataset where one car completing one race resulted in one row of performance data. 20 cars participate in each race, meaning that one race is represented by 20 rows. We included data about the weather, the tracks altitude, composition, popularity, and finally the car's on-track performance. A lack of data pushed us towards concentrating on the 1996-present era of the sport, which smoothed out some of the regulatory inconsistency issues previously discussed.

We concluded that when given out-of-sample data Bagging Classifier correctly predicted a vehicle failure ~66% of the time compared to ~63% and ~53% of the time for Extra Trees Classifier and Random Forest Classifier, respectively. Therefore, we can conclude that the Bagging Classifier is the most appropriate estimator for our use-case. Although we were impressed with our results, we were unable to move beyond model evaluation and perform hyperparameter tuning due to time constraints.

We collected data related to track conditions, track design, and car performance to be used as primary features to estimate vehicle failure

F1 will soon become significantly more financially equitable than it currently is. The sport is introducing driver salary caps, team spending caps, revenue sharing, and wind-tunnel time limits beginning in 2022

We tested a variety of binary classification models and determined that Bagging Classifier was the strongest estimator of vehicle failure, given our data

Background & Problem Identification

Formula 1, also known as F1 or the FIA World Championship, is the highest class of international auto-racing for single-seat formula racing cars sanctioned by the Fédération Internationale de l'Automobile (FIA) (The FIA, 2021). "Formula" refers to the rules and regulations which cover racing activities, car design & construction, team management, and financing. There are typically 10 Teams, referred to as Constructors, and 20 Drivers, two per team, participating in Formula 1 at any given time. These 10 Constructors and 20 Drivers travel around the world to compete weekly on different circuits ranging from Baku, Azerbaijan to Silverstone in The United Kingdom.

Formula 1 is an international racing league where teams such as Ferrari, Mercedes, McLaren, and Aston Martin design, build, then race the fastest racecars in the world on over a dozen different international tracks

A Formula 1 race is typically composed of a series of smaller races and then one large race over the course of a full weekend. Constructors and Drivers first participate in a series of mini races, called 'qualifying' to determine the order of cars at the beginning of the final race. This ordering is known as the grid. Once the grid is set, the real race begins. Races are no longer than 2 hours and cover a minimum of 190 miles of total distance over the course of several dozen laps around the track (Longman, 2021). At the end of each race, points are awarded to the Drivers in descending order, with 1st place taking the most points (usually 25) and 10th place taking the least (usually 1) (Rookie Road, n.d.). Drivers in positions 11-20 are not awarded points. Constructors inherit the point totals of their Drivers. Constructors and Drivers simultaneously compete for two related, but independent seasonal competitions. The first competition is for the World Drivers' Championship (WDC). The WDC is awarded to the single driver with the most points at the end of the season. The second competition is the World Constructors' Championship (WCC). The WCC is awarded to the team which has the most points overall at the end of the season between their two Drivers.

On average, the cost of replacing a totaled F1 Livery is $1.2M dollars. In 2021 due to COVID-19 cost controls, if a driver's vehicle had its engine replaced 3 times or more, the driver would be penalized on track by grid-deductions and time penalties, meaning that a failure to manage costs would directly impact on-track performance and results

Formula 1 is not a level playing field. Like American professional baseball, each team has an independent budget it can spend on vehicle R&D, production, part replacement, Driver salary, team salary, factory equipment, etc. Also, making the top 3 of the WDC or WCC has traditionally resulted in bonus payments for winning Constructor's next season, meaning that Constructors that perform well can continue to do so by consistently outspending their competitors. This is the essence of modern Formula 1 racing. The competition on track translates to a larger competition to acquire, manage, and efficiently use resources to produce the best factory, staffed with the best engineers, to produce the best car, to be driven by the best Drivers.

Teams do not compete on a level playing field. Most costs for driver salary, much of the livery R&D, and part replacement, all come directly out of the team's individual budget. There is little cost or revenue sharing in contrast to other major sports

With the financial considerations of the sport in mind, our team attempted to better understand vehicle failure so that we could support teams in making financial forecasts as well as making race strategy decisions. If we could better understand the probability and factors that influenced vehicle failures of all kinds, including accidents, collisions, and mechanical failure, we could help teams manage financial risk and support strategic decision-making. Our goal was to develop a model that could predict vehicle failures within a specific race for racing strategy purposes. We didn't reach this objective. However, we did make significant progress in using classification algorithms to model vehicle failure on a per-race basis, allowing us to potentially forecast the number of vehicle failures in a season and thus improve financial planning.

Because vehicle failures of all kinds, including accidents, collisions, and mechanical failure are so consequential from a competitive and financial perspective, we focused on producing estimates of car failure on a per-race basis

Hypothesis Generation

To develop an interesting and insightful hypothesis, our team reviewed the data and researched common Formula 1 issues that we could identify and give insight into. After discussion and literature review, we initially decided on two hypotheses:

- **Original Hypothesis 1:** Predict when a tyre needs to be replaced. There are many tyre types so an optimal strategy is to pit right before the tyre needs to be replaced.
- **Original Hypothesis 2:** Predict what position a car would finish in a race given starting position, global standing, weather, etc.

Unfortunately, we decided that validating either hypothesis wasn't possible with the data we have. Detailed information on tyre type and performance is closely guarded by the teams and there is not enough raw data available to do position predictions. Consequently, we shifted our hypothesis to the probability of a Driver completing the race successfully based on various factors. Specifically, we hypothesize that:

- Starting Position (Grid) has a strong direct relationship with the probability of completing the race.
- Average lap time has an inverse relationship with the probability of completing the race.
- Precipitation and higher average temperature have an inverse relationship with the probability of completing the race.

While the suppositions above appear intuitive, we chose these hypotheses because of the time frame of our project, the data available, and our interest in the monetary implications of the sport.

We originally attempted to predict vehicle failures during races, as in predicting probability of completing a lap on a per-lap basis. We had to discard this idea because of a lack of data. We ended up focusing our project on a per-race timeframe instead, because our data best fit this approach

Data Summary

Initial Ingestion

Our team drew source data from two locations. The first was a popular dataset found on Kaggle which contained several CSVs outlining data related to race outcomes, lap times, seasonal point totals, vehicle failures, Driver background, Constructor background, etc. This data ranged from the beginnings of the sport in 1950 to today. The second source of our data was the NOAA Climate Data website. We used this website to compile weather data, including temperature and precipitation, for all the race locations and dates which we used in our final dataset. Our Team assembled a complete data map describing what information we started with. That is available in **Figure *1***.

Our data came from two primary places: Kaggle and NOAA Climate Data. We have 4 primary areas of data:

1. *On track performance*
2. *Track conditions*
3. *Track features*
4. *Weather Data*

Our data has 4 major keys which connect our CSV's together. Those key ID's are Driver ID, Constructor ID, Race ID, and Circuit ID. A complete description of each of these key's is available in **Table *1***, below.

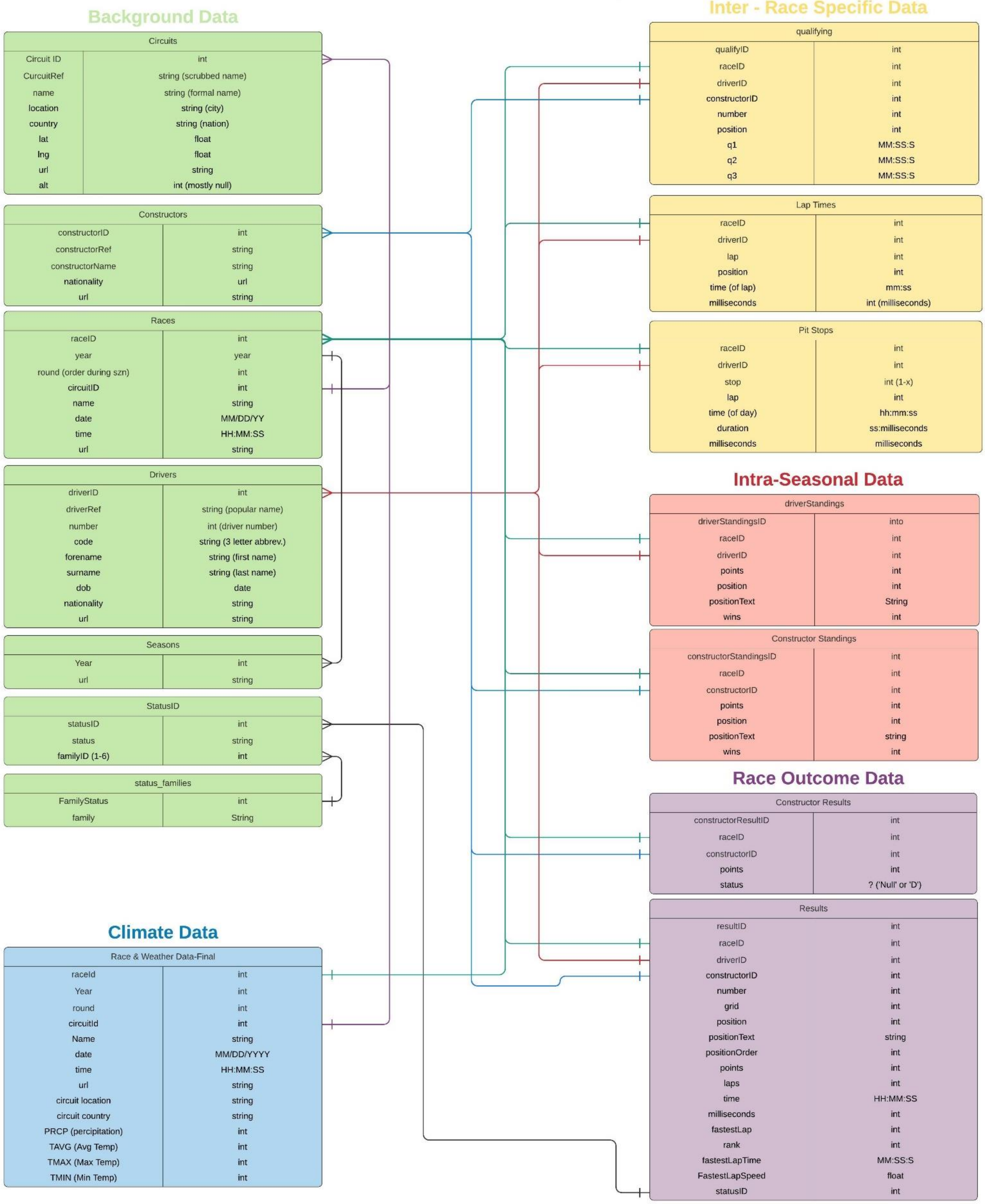
Table 1: Key IDs for Data

Driver ID	Unique identifier for the driver in the seat of the car.
Constructor ID	Unique identifier for the Constructor (team) which owns and operates the car.
Race ID	Unique identifier which states which specific race the data is related to. Unlike Circuit ID, Race ID includes both a time and place. Multiple Race IDs connect to a single Circuit ID.
Circuit ID	Unique identifier for the track the race is occurring on. There are approximately 71 unique tracks in the sport's history.

Colors are replicated from full Data Map

Constructor ID was a column we attempted to use as a feature but struggled with. F1 has 10 franchise slots, but the name and branding of those franchises changes frequently. Furthermore, because the pool of Drivers and engineers is relatively small and niche, teams can go bankrupt, exit the sport, be bought out, and reopen without experiencing particularly significant amounts of turnover or change beyond a change in ownership and branding. This begs the question, what is the definition of a new or different Constructor?

Figure 1: Project Data Map



Background Data

Circuits	
Circuit ID	int
CurcuitRef	string (scrubbed name)
name	string (formal name)
location	string (city)
country	string (nation)
lat	float
lng	float
url	string
alt	int (mostly null)

Constructors	
constructorID	int
constructorRef	string
constructorName	string
nationality	url
url	string

Races	
raceID	int
year	year
round (order during szn)	int
circuitID	int
name	string
date	MM/DD/YY
time	HH:MM:SS
url	string

Drivers	
driverID	int
driverRef	string (popular name)
number	int (driver number)
code	string (3 letter abbrev.)
forename	string (first name)
surname	string (last name)
dob	date
nationality	string
url	string

Seasons	
Year	int
url	string

StatusID	
statusID	int
status	string
familyID (1-6)	int

status_families	
FamilyStatus	int
family	String

Climate Data

Race & Weather Data-Final	
raceId	int
Year	int
round	int
circuitId	int
Name	string
date	MM/DD/YYYY
time	HH:MM:SS
url	string
circuit location	string
circuit country	string
PRCP (percipitation)	int
TAVG (Avg Temp)	int
TMAX (Max Temp)	int
TMIN (Min Temp)	int

Inter - Race Specific Data

qualifying	
qualifyID	int
raceID	int
driverID	int
constructorID	int
number	int
position	int
q1	MM:SS:S
q2	MM:SS:S
q3	MM:SS:S

Lap Times	
raceID	int
driverID	int
lap	int
position	int
time (of lap)	mm:ss
milliseconds	int (milliseconds)

Pit Stops	
raceID	int
driverID	int
stop	int (1-x)
lap	int
time (of day)	hh:mm:ss
duration	ss:milliseconds
milliseconds	milliseconds

Intra-Seasonal Data

driverStandings	
driverStandingsID	into
raceID	int
driverID	int
points	int
position	int
positionText	String
wins	int

Constructor Standings	
constructorStandingsID	int
raceID	int
constructorID	int
points	int
position	int
positionText	string
wins	int

Race Outcome Data

Constructor Results	
constructorResultID	int
raceID	int
constructorID	int
points	int
status	? ('Null' or 'D')

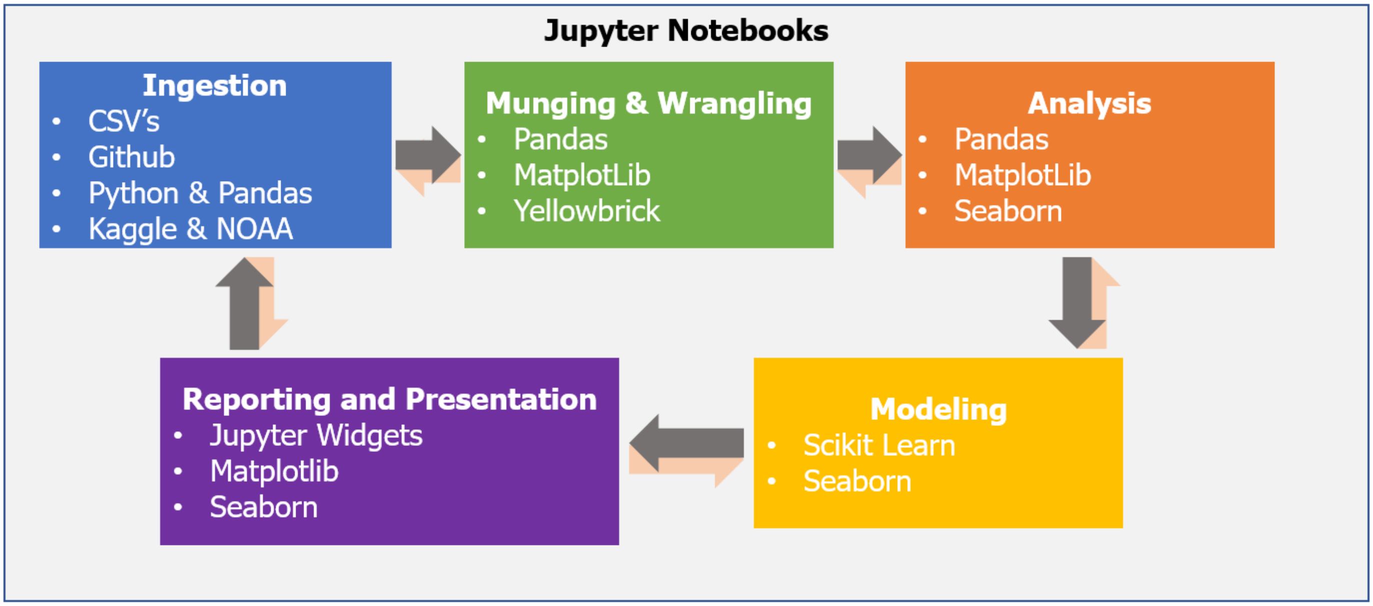
Results	
resultID	int
raceID	int
driverID	int
constructorID	int
number	int
grid	int
position	int
positionText	string
positionOrder	int
points	int
laps	int
time	HH:MM:SS
milliseconds	int
fastestLap	int
rank	int
fastestLapTime	MM:SS:S
FastestLapSpeed	float
statusID	int

Selection of Tools

Our team initially chose to use Pandas, Matplotlib, Plotly, Seaborn, Scikit Learn, and Yellowbrick as core tools. **Figure *1*** below provides a graphical description of our process and tool selection. We also initially intended to use PostgreSQL as our database. However, because the volume of data we were using was low and came originally in the form of a CSV, we elected to stick with CSVs as our storage method and used GitHub to store updated versions of our data. The process for building our final dataset was to iteratively take sections of data from our original Kaggle and NOAA datasets, blend them together, perform transformations, then upload revised versions of our dataset to GitHub in the form of a CSV. We would then load this revised CSV for modeling.

Our project architecture stuck mainly to packages and libraries in python. We did not have a large enough dataset to mandate the use of a SQL database. Most of our data came in CSVs and we stuck with that structure

Figure 2: Project Architecture



Feature Selection & Engineering

The construction of our final dataset started with 'results.csv'. This CSV covers the outcomes of all F1 races within its domain. It states the starting and finishing position of each car as well as aggregate race times. It also provides for things like fastest lap and average speed. Most importantly, this CSV contains a status column which prescribes the status of the car at the end of the race. Status originally could be any of 138 unique codes. Each code represented a unique mechanical failure or scenario which could derail a Driver's race, from steering wheel failure to multi-car collisions. Status also denotes if a car successfully completes a race without issue. For our purposes, we transformed the status column to be a binary column simply confirming whether a single car completed all laps in a race. By starting with

Our data was heavily based on the 'results.CSV' component of our original Kaggle dataset. We iteratively added additional data to this CSV and transformed features to produce our final dataset, which was saved as a CSV and then loaded for modeling results.csv, we made the first key decision of our project. We lacked sufficient data to prescribe the status of a car on a per-lap basis. This meant that we couldn't construct a model to support race-strategy decisions or prescribe the risk a car faced of failure on a per-lap basis. Instead, we zoomed out to a per-race basis and focused our project in that area. As a result, each row of our final dataset represents the result of a single car in a single race.

Exploration of our data identified a second key issue and decision point, which we were able to turn into a positive note. Our dataset had an enormous hole in it running from the mid-70s to the mid-90s. During this period, we do not have any data regarding each car's lap times, pit times, or any other on-track performance data. We do know if these cars completed their race or not. However, we know little else. We elected to focus our project on what we did have, which mainly ran from 1996-present. This ended up working in our favor because F1 has become a significantly more stable sport during this period. From 1996-present, there are comparatively fewer rule changes or Constructor's entering and exiting the sport, unlike prior eras. Although the cars have seen strong performance improvements over the period, the bulk of the ruleset has not changed extensively during this period. We believe that this design choice ultimately made our dataset more viable and practical to be used for machine learning purposes, as we were focusing on the sport during its prime. These two decisions framed what data was ultimately used for our dataset. **Table *2*** below provides a detailed description of each column as well as transformations performed on that data.

The structure and composition of our Kaggle dataset forced two key decisions in our project:

1. *We oriented our timeframe to be per-race because we lacked sufficient data to prescribe a per-lap status to each car. Also, a per-lap approach produced an enormous class imbalance which we thought to be too much even with sampling techniques*
2. *We adjusted our dataset to only include data from 1996-present because of a large gap in lap times prior to the mid 1990's*

Table 2: Data Descriptions

X or Y & Count	Column Name	Description of Data & Transformations Performed	Data Type	Data Source
N/A	4 Major Key ID's	We chose to leave our 4 major Key IDs in our dataset in case we wanted to add additional data. They were not used directly as features though because they have no meaning if left in their numeric form.	INT	Kaggle
Y	Completion Status	A reduced version of status from results.csv, Completion Status defines if a car finished a race successfully (1) or not (0). Three key transformations were made: 1. Any cars that failed because of a disqualification or other niche error were discarded, because our data is not descriptive of disqualifications or other niche issues.		**Kaggle** Results.csv

X or Y & Count	Column Name	Description of Data & Transformations Performed	Data Type	Data Source
		2. Cars that failed due to mechanical error on Lap-0, the start of the race, were also withdrawn from the data, because these events are not our target. 3. Because a strong majority of our cars complete the race, we used Synthetic Minority Oversampling Technique (SMOTE) to address our class imbalance issue prior to modeling.		
X1	Grid	Grid defines the starting position of a car on the racetrack. We did not transform Grid and left it as continuous. We selected this feature because we strongly suspect that cars closer to the front face significantly reduced risk at the start of each race, while cars in the back are much more likely to become collateral casualties to accidents or collisions ahead of them.	Int (1-24)	**Kaggle** Results.csv
X2	Year	Year defines the year that the race occurred in. This was our time feature, and we did not perform any transformations on it. We selected this feature because we wanted to include time if the sport was becoming safer or more dangerous season-over-season.	Int (1996–2021)	**Kaggle** Results.csv
X3	Alt. (Altitude)	Altitude defines the average altitude of the circuit being raced on. This feature was normalized using a log(x+1-min(x)) transformation technique. We selected this feature because we suspect that changes in altitude affect car/driver performance and therefore impact risk of vehicle failure or accident.	Int	**Kaggle** Races.csv
X4	isHistoric	This is a binary feature that describes tracks that are 'historic' to the sport. Historic tracks are not formally designated as such by the FIA. However, these tracks are visited almost every single year with very rare exception by the sport.	Boolean	**Excel**[1] Circuits.csv

[1] Because there is no technical definition of isHistoric, we hand-gathered this data from online and prescribed whether a track was historic by opening the CSV and marking down for each track their status. This wasn't an ideal solution but worked with the time and capabilities we had and was practical for how few data points we were generating.

X or Y & Count	Column Name	Description of Data & Transformations Performed	Data Type	Data Source
		We thought this feature was significant because we expect teams to build and model their cars around these tracks first. Likewise, drivers will practice in simulators on these tracks as they prepare for the season.		
X5	Average Lap Time	This is a numeric feature describing the average lap time of a car. All cars have an average lap time so long as they completed one lap. We normalized this feature using a method we developed which normalized by RaceID, accounting for changes to the track and the league's average performance. We also made a log(x) transformation after normalizing. We also used imputation to address several outliers. We selected this feature because we wanted to understand if relatively faster or slower cars were more or less likely to experience failure.	Float	**Kaggle** Results.csv
X6	Minimum Lap Time	Minimum Lap Time is a numeric feature that describes the fastest single lap the car completed during the race. Like average lap time, we normalized this feature by building our own function which normalized by RaceID and removed major outliers using simple imputation. We selected this feature because all teams and drivers are vying for the fastest lap at the end of races when their cars are lightest, due to a loss in fuel, and their tyres are most worn down, due to the rigors of the race. We suspect that in these moments, the drivers that push their cars to the limit are putting themselves at enhanced risk.	Float	**Kaggle** Results.csv
X7	PRCP (Precipitation)	Precipitation is a numeric feature that describes the amount of rain which occurred on the day of the race. Critically, it should be noted that this is an estimate, and not an exact figure. We selected the nearest weather stations to each circuit, so the data is not exact. This feature was transformed using log(x+1). Although this variable has outliers, we chose not to replace them using imputation because they weren't as extreme. We selected this feature because rain significantly impacts the ability of cars to grip the track and stay on course. Rain also impacts visibility.	Float	**NOAA** Race & Weather Data – Final.csv

X or Y & Count	Column Name	Description of Data & Transformations Performed	Data Type	Data Source
X8	TAVG (Average Temperature)	Average Temperature is a numeric feature that describes the average temperature in Fahrenheit during the race. Like PRCP, this data is an estimate from the nearest weather station to the track. We selected this feature because extreme temperatures, both high and low, can have major implications on car performance and specifically brake performance.	Float	NOAA Race & Weather Data – Final.csv
X9	Track Type	Track Type describes the type of track that is being raced on. There are 2 kinds of tracks in our data: 1. Racetracks (0) – Tracks specifically built for and only used for auto-racing. 2. Street tracks (1) – Racetracks that are constructed entirely from public roads which are repaved and temporarily used for racing. These tracks are usually in the heart of major cities. We selected this feature because these different types of tracks have very different demands on both cars and drivers, with street tracks being narrower and placing more emphasis on turns relative to racetracks which emphasize straight-line speed.	Boolean	Excel[2] Circuits.csv
X10	Binned Circuits	Binned Circuits is a categorical column we built to describe the relative popularity of each circuit. We grouped the circuits in our data into 6 groups based on the total number of times each circuit was visited. Binned Circuits was transformed into a series of binary columns using One Hot Encoding prior to being loaded for modeling. We selected this feature because we strongly suspect that driver familiarity with a circuit impacts performance.	Categorical (1-6)	Kaggle Results.csv

Total Columns: 22	**Total Rows:** 9258

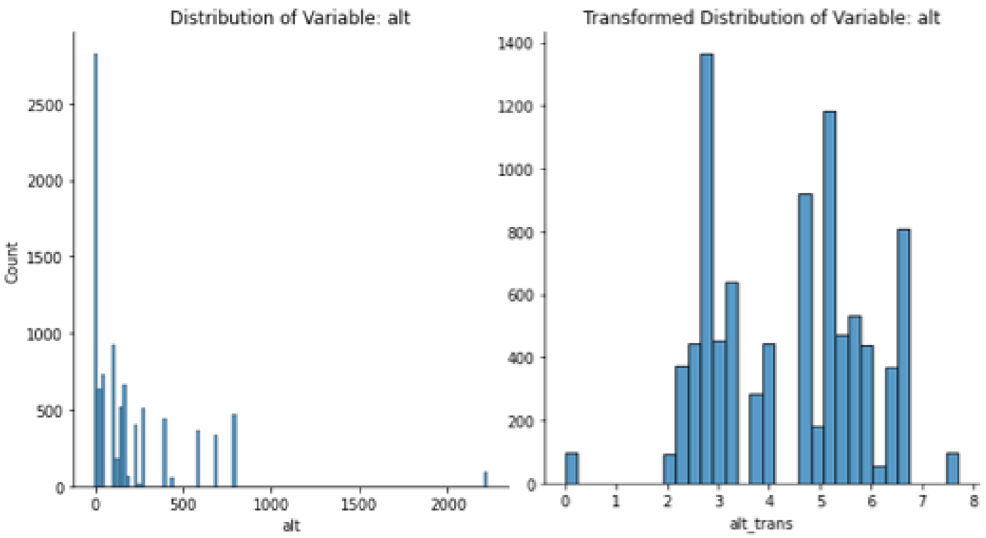
[2] Similar to isHistoric, we marked this down manually within the CSV by looking up the tracks online

Column Shape, Composition, and Impact of Transformations

Altitude

As seen in **Figure *3*** below, altitude was positively skewed, so we used a log(x + 1 - min(x)) transformation to create a more normalized feature, seen in the second distribution plot.

Figure 3: Original vs. Transformed Altitude



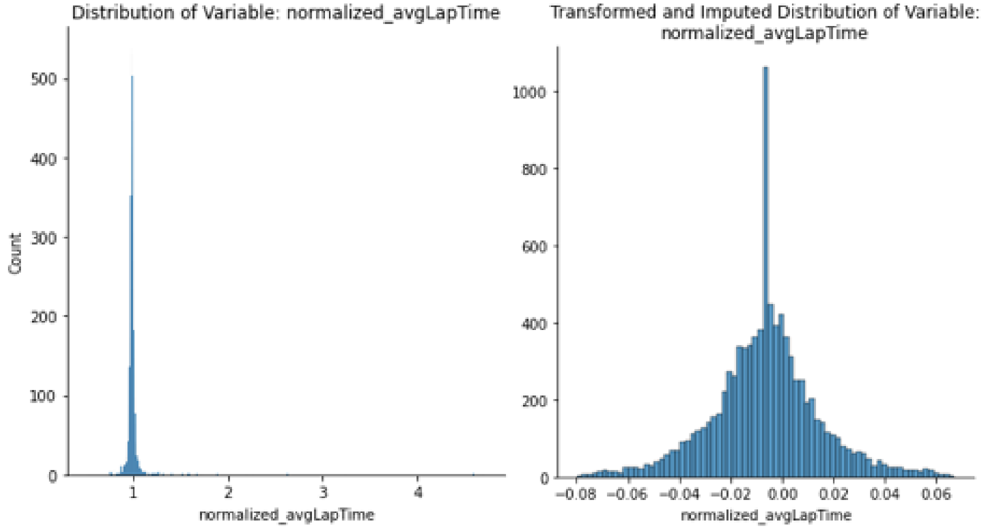
Average & Minimum Lap Time

To summarize, we adjusted average lap time in three ways:

- We normalized them by raceId to account for seasonal change
- We used a log(x) transformation to standardize the distribution
- We used simple imputation by the median to address outliers.

These new and old distributions are shown in **Figure 4** below.

Figure 4: Original vs. Transformed Average Lap Time



Altitude had numerous nulls in our initial dataset. However, when we excluded all years where we lacked lap times, that also dropped all the null values from our final dataset

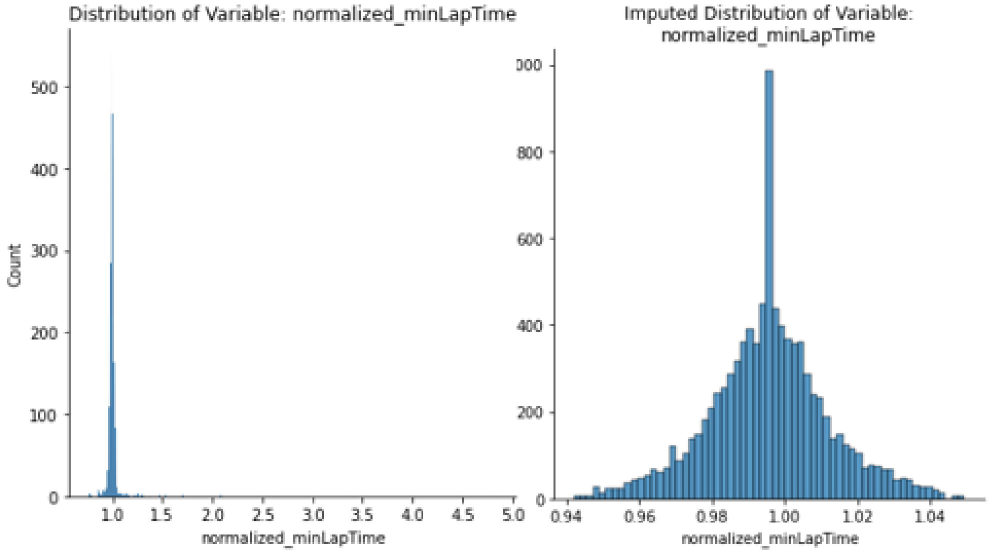
Because the log(x + 1) method "add[s] an arbitrary constant to the data", there has been some critique of it (Wicklin, 2011)*, but we believed it to be the best method for transforming any of our features that had negatives and/or zeros, like altitude. We also don't believe it interfered with the feature's relationship to our target variable. We used the same method while transforming precipitation*

We classified outliers as the following: Q1 = Quartile 1, Q3 = Quartile 3, IQR = Interquartile Range. Outliers that fell below Q1 - 2.5 x IQR or above Q3 + 2.5 x IQR were replaced with the median. The large number of outliers was a cause for concern, but we wanted to prioritize normalizing the distribution

Like average lap times, we made the same changes for minimum lap time apart from the log(x) transformation. These new and old distributions are shown in **Figure *5*** below.

Much of the imputation went the same way as it did with the average lap times, except for the outliers, which we defined as being below Q1 - 2.0 x IQR or above Q3 + 2.0 x IQR. Once again, we prioritized the normalization of the feature's distribution

Figure 5: Original vs. Transformed Minimum Lap Time



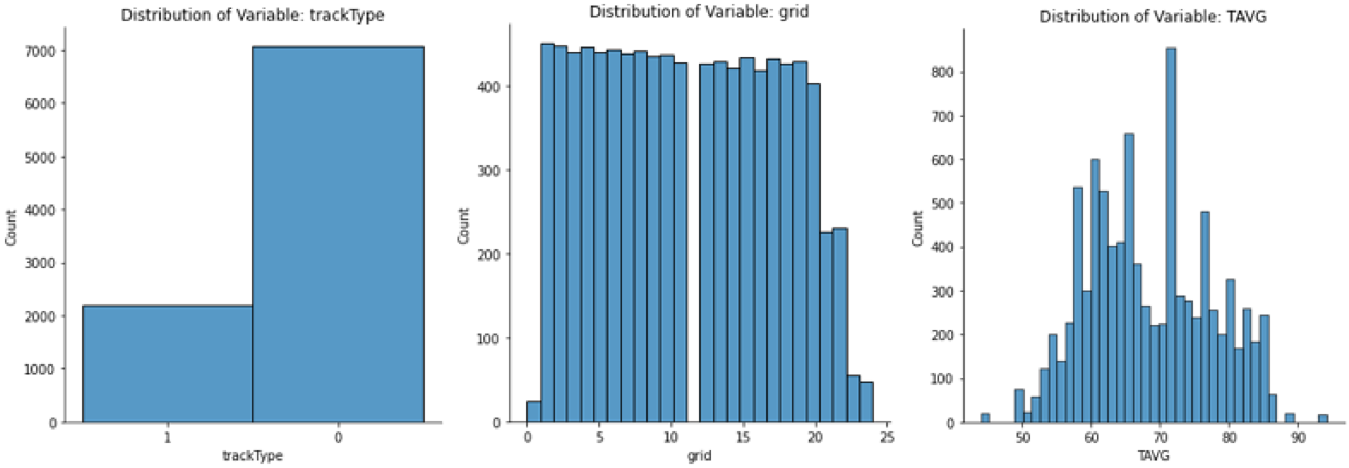
Distributions of other Key Variables

We also produced plots of our grid, track type, and average temperature during the EDA phase of our project. Those plots are available in **Figure *6*** below. A few notes on these:

1. Although we were not able to resolve the issue in production, there is no gap in grid. We have a roughly equal amount of grid placements at 11 as any other position.
2. Grid moves slightly past 20, which is the theoretical maximum number of cars. This is because Drivers can be penalized for driving errors during qualification or the formation lap and be relegated to start the race from the pitlane. These scenarios were encoded as grid positions 22-24.

The track Type feature began with three options of race, road, and street, of which we assigned the values 0, 1, and 2, respectively. When we narrowed our data down to the 1996 to present time frame, however, we found that there were no longer any road tracks. As such the options are just race and street

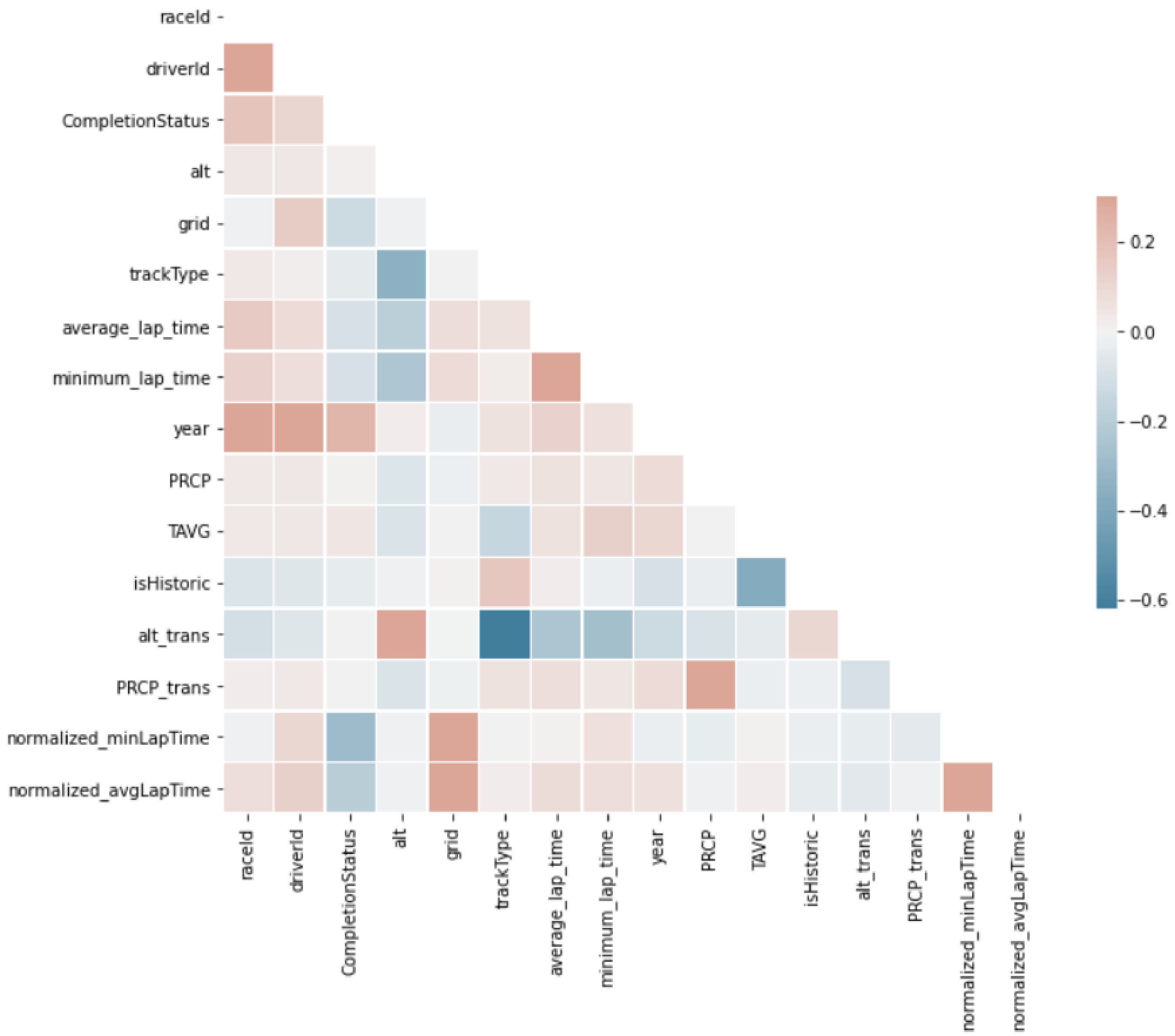
Figure 6: Distribution of other Key Features



Correlation Matrix

Our team produced a correlation matrix during exploratory data analysis, seen in **Figure *7*** below. Note that the correlation of any variable is generally very low. For this reason, we elected not to spend time automating feature selection or incorporating feature selection processes in our code.

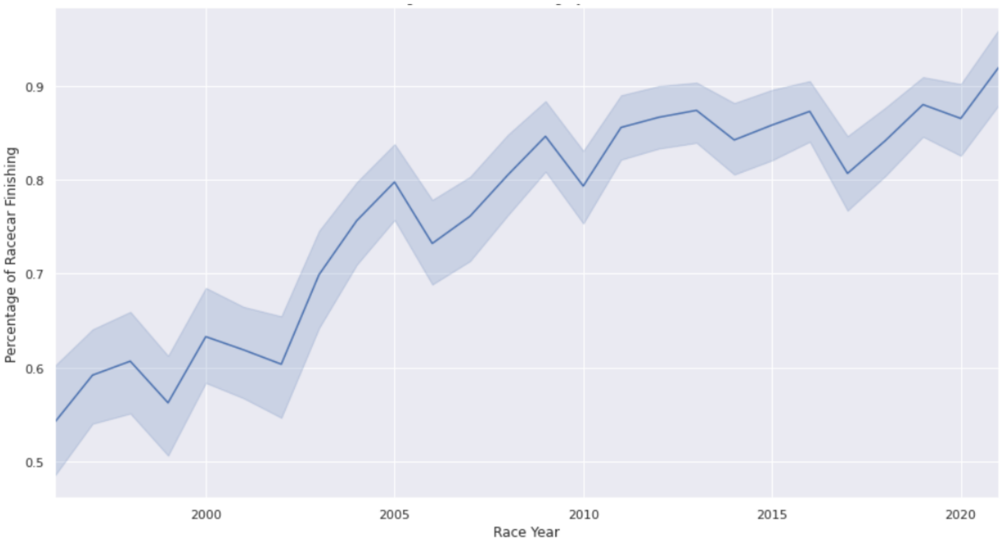
Figure 7: Correlation Matrix



Formula 1 Safety

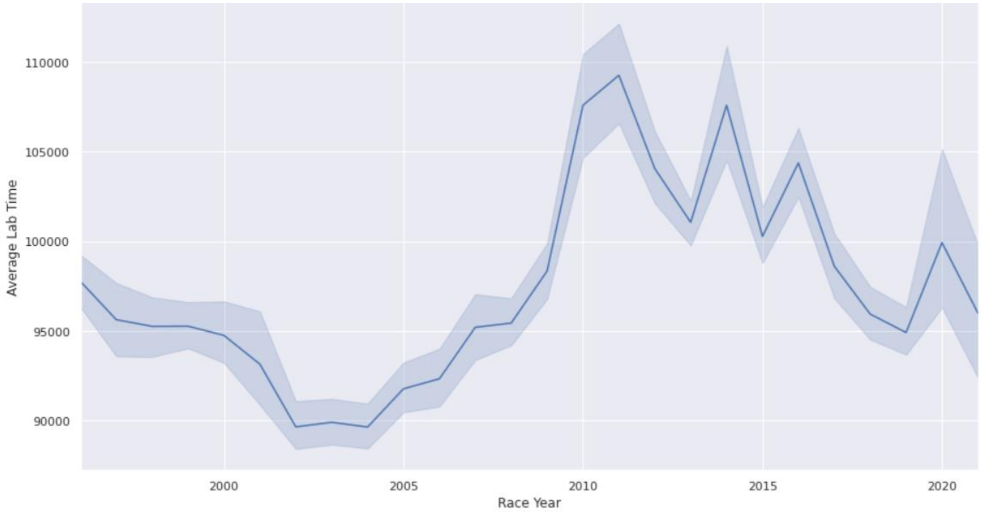
A critical bias at play in all our data is the constantly evolving ruleset of Formula 1. Two major goals of the sport, driver safety, and economizing team costs, have pushed the sport towards cars that are heavier, safer, but much slower. This has resulted in a steady increase in consistency in cars finishing, but a loss in net lap time. This is observed in **Figure *8*** and **Figure *9***, respectively.

Figure 8: Percentage of Cars Finishing by Year



Formula 1 is a constantly evolving sport. As such, there is a significant amount of variation in car performance year over year. These two plots highlight the increased safety and consistency of cars, but also the resulting loss in raw speed due to an increase in weight. Therefore, normalizing by year and track was very important for our preprocessing

Figure 9: Average Lap Time by Year



Modeling

We focused on using binary classification models to estimate vehicle failure because our data was labeled, and we had a binary target. We evaluated our data using:

- Support Vector Classification (SVC),
- Nu-Support Vector Classification (NuSVC),
- Linear Support Vector Classification (LinearSVC),
- Stochastic Gradient Descent (SGD) Classifier,
- K-Neighbors Classifier,
- Logistic Regression,
- Logistic Regression using Cross Validation (CV),
- Bagging Classifier,
- Extra Trees Classifier, and
- Random Forest Classifier.

Because our data is labeled and we are seeking to predict a binary outcome, our project was definitionally a supervised learning exercise and we focused on implementing binary classification models

We started with several basic features such as average lap time, and progressively expanded our feature space and preprocessing, testing against all the major classification models. **Table 3** describes our highest F1 Scores after installing our strongest munging and wrangling framework. We discarded all but the four best models during our process because we consistently achieved F1 scores that were too high to be practical. We discarded all but the four best models during our process.

Because this is the first time most members of our team have attempted to perform a machine learning exercise, we are somewhat skeptical of our results. We struggled with a data leakage issue throughout our project which resulted in our models overfitting and performing suspiciously well on our test data. Although we made multiple changes to our code to address this leakage, we do suspect that it still at least partially exists

Table 3: Final Evaluation Results

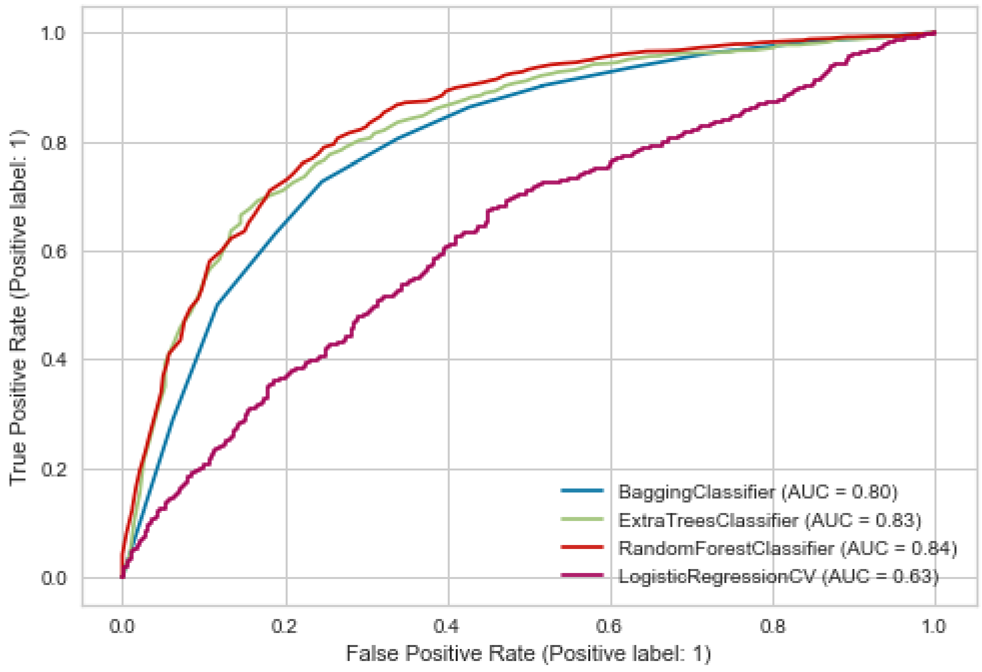
Model Name	F1 Score
Logistic Regression CV	0.8342632097144733
Bagging Classifier	0.8448212311094729
Extra Trees Classifier	0.8811813186813187
Random Forest Classifier	0.8881094352858645

Results & Conclusion

Although Bagging Classifier was strong and we were surprised with our results, we were unable to move beyond model evaluation and perform hyperparameter tuning on any of the 4 best models because these scores represented the ceiling of our capability at the time. We worked with this result to produce a collection of ROC plots, seen in **Figure *10*** below to determine which model was best for our use-case.

After implementing our most sophisticated preprocessing work, we received the following results, which initially indicated that Random Forest Classifier was the strongest model overall for predicting race outcomes.

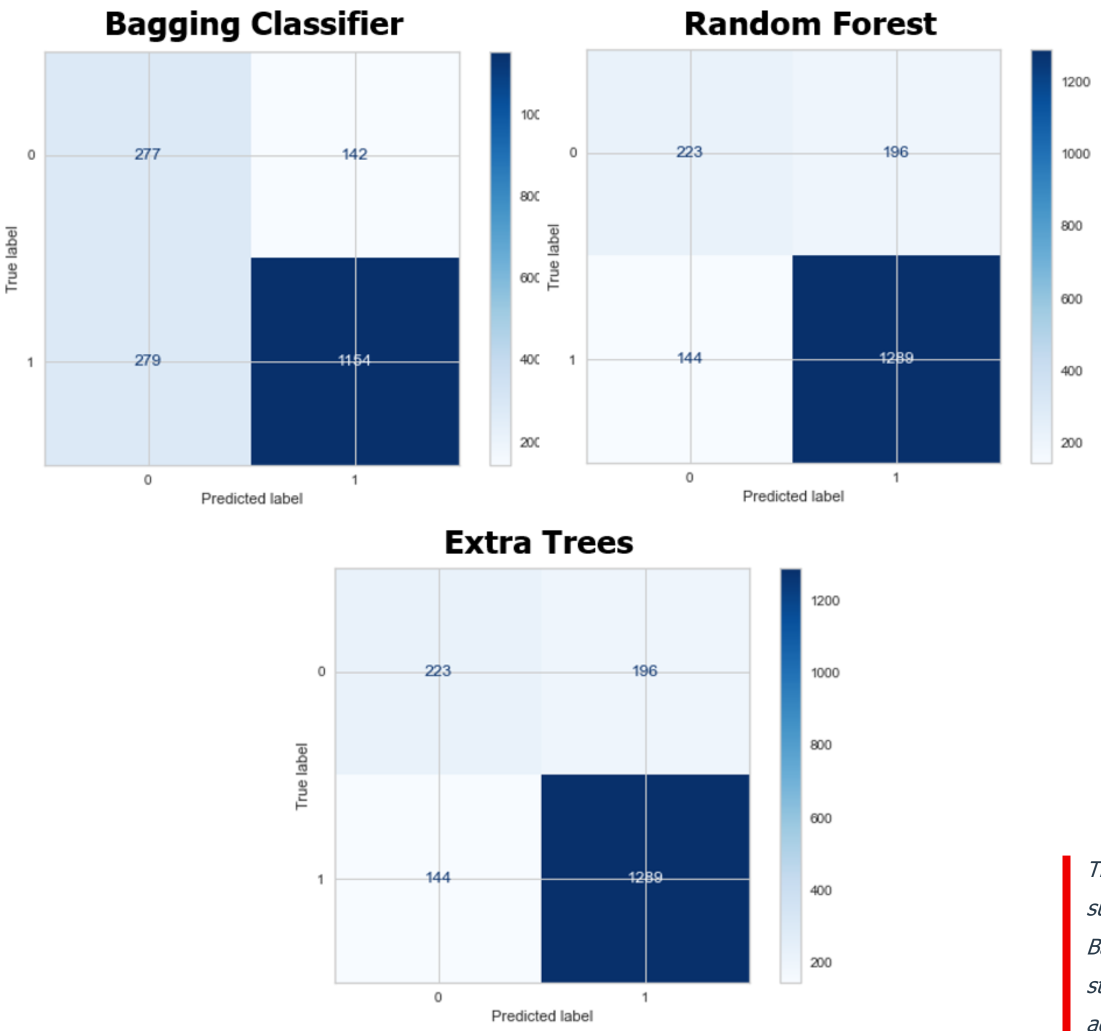
Figure 10: ROCAUC Curves for Classification Report



The shape of the ROC Curves of Random Forest Classifier and Extra Trees Classifier appear best for predicting binary outcomes. They also have high AUC scores, suggesting that they cover the majority of scenarios

Bagging Classifier, Extra Trees Classifier, and Random Forest Classifier perform best at distinguishing our target classes as they hug the upper left corner of the plot and have high AUC scores of .84, .83, and .80 respectively. We can conclude that these three models are most viable. For our business case, we prefer a model best suited to predict true negatives (car failures). To get a better sense of which model was best at predicting true negatives, we plotted the confusion matrices of these 3 models. That result is depicted in **Figure 11** below.

Figure 11: Confusion Matrices for 3 Strongest Models



The Confusion Matrices support the argument that Bagging Classifier is the strongest model because it accurately predicted the largest proportion of vehicle failures of the three

These matrices confirm that Bagging Classifier correctly predicts the greatest number of vehicle failures. We also concluded that Bagging Classifier was the strongest overall estimator for both classes based on Specificity:

Bagging Classifier:

(TN / (TN + FP)) = 0.6610978520286396

Extra Trees Classifier:

(TN / (TN + FP)) = 0.5322195704057279

Random Forest Classifier:

(TN / (TN + FP)) = 0.6252983293556086

We concluded that when given out-of-sample data Bagging Classifier correctly predicted a vehicle failure ~66% of the time compared to ~63% and ~53% of the time for Extra Trees Classifier and Random Forest Classifier respectively. Therefore, we can conclude that the Bagging Classifier is the most appropriate estimator for our use-case.

Next Steps & Unanswered Questions

Unfortunately, every project is limited by time which necessitates setting realistic project objectives. We identified five improvements that would significantly improve the performance of our models that we did not implement due to a lack of time and know-how. The team believes these future enhancements would improve the overall project value.

1. Use a PostgreSQL database to store all data. This would enhance the data integrity and accessibility as compared to storing the data in CSVs on GitHub.
2. Develop a multi-class model to assess the probability of specific types of vehicle failure, such as mechanical issues, collisions, accidents, etc. This would improve the specificity of our predictions greatly.
3. Use ensembling to improve the F1 score.
4. Develop a web application for hosting our final widget. The team discussed using Django to enable users to predict in real time whether a racecar would complete a race given certain conditions.
5. We would like to integrate two application program interfaces (APIs) to directly pull data from the Formula 1 website (for race data) and the NOAA website (for weather data). Both APIs coupled with a SQL database would enable automated updates to model predictions.
6. For increased model specificity performance we can increase the threshold for predicting vehicle failure (0) while sacrificing the performance of accurately predicting finishes (1).
7. Producing a more specific model at the risk of sacrificing sensitivity is a risk that is well worth it when considering the business-case of our project.
8. With driver safety and potential financial risk in mind, we would rather produce a model more prone to predicting that a driver would crash, when in reality they might have actually finished.
    - We prefer this type of error to the alternative, which is producing a model more prone to predicting that a driver would finish, when in reality they actually crash/do not finish.
    - Another future implication of these results would be to construct a robust data preprocessing pipeline that includes every transformation that we have performed on our data prior to modeling.

Our project did not move past the model evaluation process. We did not perform any hyperparameter tuning or implement an ensemble model, which was our goal for this phase of work. We propose this and several changes as keys for future development.

  - This would be especially useful to standardize unprocessed, out-of-sample data in an efficient way to be prepared for modeling.

9. Lastly, hyperparameter tuning of our most viable models to optimize model performance would be a beneficial next step for this project. Using hyperparameter tuning to achieve higher model performance would be crucial if we were to pitch this project in a real-world scenario with high financial stakes and driver safety on the line.

Appendix

Literature Review

In the eBook "ACCELERATING THE FAN EXPERIENCE: How FORMULA 1 is driving the future of racing using machine learning and AWS," Formula 1 describes how they teamed up with AWS to prioritize the fan experience and make the sport more entertaining. The book starts with a discussion on all the changes that Formula 1 has already gone through, from its conception in the early 1950s to the sport that it is today (Formula 1; Amazon Web Services (AWS), 2021). Formula 1 has done its best to create both faster and safer cars, touching on things like "vehicle body designs and aerodynamics, body materials, tyre compounds, engine builds," and more (Formula 1; Amazon Web Services (AWS)). Most recently, Formula 1 partnered with AWS to utilize machine learning for two specific projects: Pit Strategy Battle and the Battle Forecast. The former, as its name suggests, focuses on the strategy that comes with pitting, particularly regarding when there are two "rival drivers" (Formula 1; Amazon Web Services (AWS))**.** The model looks at "the predicted gap after their respective pit stops, and the percentage chance of an overtake" (Formula 1; Amazon Web Services (AWS))**.** Battle Forecast, on the other hand, "provide[s] insights into developing driver battles" (Formula 1; Amazon Web Services (AWS))**.** These don't necessarily have to look at pitting, like they do with Pit Strategy Battle; they can look at any battle on the track. Both projects have been developed to the point that they can make their predictions in real time.

Formula 1 is not only looking to better the fan experience through predictions, but also in making the sport itself more interesting according to what the fans want. Rob Smedley, Formula 1's Director of Data Systems, was quoted saying that "the biggest thing that everybody wants is more wheel-to-wheel racing" (Formula 1; Amazon Web Services (AWS))**.** The sport is trying to make this happen by reducing the downforce that keeps a car on the track without sacrificing any safety measures. They tested this through Computational Fluid Dynamics with the help of Amazon HPC and worked to reduce the amount of "dirty air" behind a car with the help of AWS (Formula 1; Amazon Web Services (AWS), 2021). Overall, Formula 1 and AWS have made huge steps forward regarding changing the sport, not just with the technology or physicality of the cars, but also with the addition of machine learning to make real time predictions on Pit Strategy Battle and the Battle Forecast.

Lessons Learned

The biggest lesson we learned the hard way during this project was to do your own exploratory data analysis (EDA) and not trust the work of others. Kaggle scores submitted datasets based on the number of null values in the dataset. Our data had an almost perfect score on Kaggle and passed the eye test. As such, we did not do extensive validation to confirm what data was present. However, our data replaced null values with "\N", meaning that the cell was functionally null, but counted as filled by Kaggle. This trick made us blind to several issues. As a result, we realized late in the project schedule that we were missing lap times for over two decades of total racing.

> *The single biggest disappointment with our data was the fact that we lacked tyre data. There are 3 kinds of tyres used on race day and timing pitstops relative to tyre type is the essence of race strategy. This data is clearly well guarded by team's and the league because it was not available anywhere online or via API*

References

Formula 1; Amazon Web Services (AWS). (2021, April 21). Accelerating the Fan Experience, How Formula 1 is driving the future of racing using machine learning and AWS. Seattle, Washington, United States of America.

Longman, W. (2021, June 30). *How many Laps does Each Formula 1 Race Have?* Retrieved from motorsportickets.com: https://motorsporttickets.com/blog/how-many-laps-does-each-formula-1-race-have/

Cover Photo - Morlidge, M. (2020, June 24). *F1 2020: Reverse-grid qualifying races unlikely as Red Bull say Mercedes are opposed to 'variable'.* Retrieved from Sky Sports: https://www.skysports.com/f1/news/12433/11998633/f1-2020-reverse-grid-qualifying-races-unlikely-as-red-bull-say-mercedes-are-opposed-to-variable

Rookie Road. (n.d.). *How does Scoring Work in Formula 1?* Retrieved from Rookie Road: https://www.rookieroad.com/formula-1/how-does-scoring-work/

The FIA. (2021). *Fia.gov*. Retrieved from https://www.fia.com/: https://www.fia.com/

Wicklin, R. (2011, April 27). *Log transformations: How to handle negative data values?* Retrieved from SAS Blogs: https://blogs.sas.com/content/iml/2011/04/27/log-transformations-how-to-handle-negative-data-values.html