Preprocess\_JoinFiles\_10Nov19.R

tylercampbell

2019-11-21

#############################################################################  
  
#### Introduction ####  
  
## Title: F1 Preprocessing - Join Source Data  
## Author: Tyler Campbell  
## Date: Nov 2019  
  
## IMPORTANT:  
 # Run this script after running 'Preprocess\_IndividualFiles\_10Nov19'  
 # If running full script (and not wanting to overwrite files), avoid 'Write files' section at bottom  
  
## Description:  
# This R script performs joining and cleaning of processed F1 source data  
# Sections are arranged in reverse-alphabetical order and contain section-specific notes  
# In future iterations, two joins will be performed:  
 # 'Historical' - maximizes results back to 1950, but excludes newer measures such as fastest lap data  
 # 'Modern' - includes newer measures, but excludes records before 2011  
  
#############################################################################  
  
  
  
  
#### Markdown ####  
  
# ignore this section for normal processing  
# load blank workspace when compiling/knitting  
# extra dots also are added to file paths for compilation purposes, remove when running script  
load("../metadata/workSpace\_JoinFiles.RData")  
  
  
  
  
#### Load packages ####  
  
library(tidyverse)

## ── Attaching packages ─────────────────────────────────────── tidyverse 1.2.1 ──

## ✔ ggplot2 3.2.1 ✔ purrr 0.3.0  
## ✔ tibble 2.1.3 ✔ dplyr 0.8.3  
## ✔ tidyr 1.0.0 ✔ stringr 1.4.0  
## ✔ readr 1.3.1 ✔ forcats 0.4.0

## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()

library(lubridate)

##   
## Attaching package: 'lubridate'

## The following object is masked from 'package:base':  
##   
## date

library(chron)

##   
## Attaching package: 'chron'

## The following objects are masked from 'package:lubridate':  
##   
## days, hours, minutes, seconds, years

#### Join Data Frames ####  
  
## Historical ---------------------------------------------------------------  
  
  
# This join focuses on maximizing historical data at the expense of newer measures such as pitStops and lapTimes  
 # (i.e. excludes data such as pitStop times which are only recorded beginning in 2011)  
  
  
  
# read in country Codes  
countryCodes <- read.csv("../Raw Source Data/countryCodes.csv", stringsAsFactors = FALSE)  
  
# manually correct errors caused by accent marks  
countryCodes[1214, 1] <- "Monegasque"  
  
  
# join countryCodes to drivers and constructors to convert nationality to country  
drivers <- drivers %>%  
 left\_join(countryCodes, by = c("driver\_nationality" = "Nationality")) %>%  
 select(-driver\_nationality) %>%  
 rename("driver\_homeCountry" = "Country")  
  
constructors <- constructors %>%  
 left\_join(countryCodes, by = c("constructor\_nationality" = "Nationality")) %>%  
 select(-constructor\_nationality) %>%  
 rename("constructor\_homeCountry" = "Country")  
  
  
# manually correct errors cause by compound nationalities (e.g. 'East German') or similar causes  
drivers[496, 5] <- "Italy"  
drivers[578, 5] <- "Italy"  
drivers[714, 5] <- "Germany"  
drivers[715, 5] <- "Germany"  
drivers[718, 5] <- "Germany"  
  
constructors[100, 4] <- "Belgium"  
constructors[146, 4] <- "Germany"  
  
  
# join countriesLatLong to drivers to obtain lat, long coordinates  
# source: https://github.com/knowitall/chunkedextractor/blob/master/src/main/resources/edu/knowitall/chunkedextractor/demonyms.csv  
countriesLatLong <- read.csv("../Raw Source Data/countries\_LatLong.csv", stringsAsFactors = FALSE)  
  
drivers <- drivers %>%  
 left\_join(countriesLatLong, by = c("driver\_homeCountry" = "name")) %>%  
 rename("driver\_lat" = "latitude",  
 "driver\_long" = "longitude",  
 )  
  
# clean missing lat, long for "Rhodesia"  
# arrange rows by driver\_homeCountry  
drivers <- drivers %>%  
 arrange(driver\_homeCountry)  
  
# manually update missing lat, longs  
drivers[445:448, 6] <- -19.0154381  
drivers[445:448, 7] <- 29.1548576  
  
  
# join countriesLatLong to constructors to obtain lat, long coordinates  
# source: https://github.com/knowitall/chunkedextractor/blob/master/src/main/resources/edu/knowitall/chunkedextractor/demonyms.csv  
constructors <- constructors %>%  
 left\_join(countriesLatLong, by = c("constructor\_homeCountry" = "name")) %>%  
 rename("constructor\_lat" = "latitude",  
 "constructor\_long" = "longitude",  
 )  
  
# manually update missing lat, longs  
constructors[93, 5] <- -19.0154381  
constructors[93, 6] <- 29.1548576  
  
  
# join source tables beginning from 'results' in reverse alphabetical order  
# confirm primary key  
results %>%  
 count(resultId) %>%  
 filter(n > 1)

## # A tibble: 0 x 2  
## # … with 2 variables: resultId <int>, n <int>

# join status  
resultsHistorical <- results %>%  
 left\_join(status, by = "statusId") %>%  
 select(-statusId)  
  
  
# exclude seasons  
  
  
# join races  
resultsHistorical <- resultsHistorical %>%  
 left\_join(races, by = "raceId") %>%  
 select(-race\_name, -race\_year, -race\_time, -race\_url)  
  
  
# exclude qualifying  
  
  
# exclude pitStops  
  
  
# exclude lapTimes  
  
  
# join driverStandings  
# confirm primary key  
driverStandings %>%  
 count(raceId, driverId) %>%  
 filter(n > 1)

## # A tibble: 0 x 3  
## # … with 3 variables: raceId <int>, driverId <int>, n <int>

resultsHistorical <- resultsHistorical %>%  
 left\_join(driverStandings, by = c("raceId", "driverId")) %>%  
 select(-driverStandingsId)  
  
  
# join drivers  
resultsHistorical <- resultsHistorical %>%  
 left\_join(drivers, by = "driverId") %>%  
 select(-driverId, -driver\_url)  
  
  
# join constructorStandings  
# confirm primary key  
constructorStandings %>%  
 count(raceId, constructorId) %>%  
 filter(n > 1)

## # A tibble: 0 x 3  
## # … with 3 variables: raceId <int>, constructorId <int>, n <int>

resultsHistorical <- resultsHistorical %>%  
 left\_join(constructorStandings, by = c("raceId", "constructorId")) %>%  
 select(-constructorStandingsId)  
  
  
# join constructors  
resultsHistorical <- resultsHistorical %>%  
 left\_join(constructors, by = "constructorId") %>%  
 select(-constructor\_url)  
  
  
# join constructorResults  
# confirm primary key  
constructorResults %>%  
 count(raceId, constructorId) %>%  
 filter(n > 1)

## # A tibble: 2 x 3  
## raceId constructorId n  
## <int> <int> <int>  
## 1 75 7 2  
## 2 903 3 2

# remove duplicate records  
constructorResults <- constructorResults[-c(9632, 10355),]  
  
resultsHistorical <- resultsHistorical %>%  
 left\_join(constructorResults, by = c("raceId", "constructorId")) %>%  
 select(-constructorResultsId, -constructorId)  
  
  
# join circuits  
resultsHistorical <- resultsHistorical %>%  
 left\_join(circuits, by = "circuitId") %>%  
 select(-circuitId, -circuit\_url, -circuit\_lat, -circuit\_long)  
  
  
  
  
#### Clean Newly Merged Data Frame ####  
  
# consider converting points earned to boolean  
# consider creating a normalized column of % of total laps finished in previous race round  
  
# inspect df  
glimpse(resultsHistorical)

## Observations: 24,104  
## Variables: 34  
## $ resultId <int> 1, 2, 3, 4, 5, 6,…  
## $ raceId <int> 18, 18, 18, 18, 1…  
## $ result\_startingGridPosition <int> 1, 5, 7, 11, 3, 1…  
## $ result\_finishOrder <int> 1, 2, 3, 4, 5, 6,…  
## $ result\_pointsEarned <int> 10, 8, 6, 5, 4, 3…  
## $ result\_lapsCompleted <int> 58, 58, 58, 58, 5…  
## $ result\_finishTimeMillisec <int> 5690616, 5696094,…  
## $ result\_fastestLap <int> 39, 41, 41, 58, 4…  
## $ result\_fastestLapRank <int> 2, 3, 5, 7, 1, 14…  
## $ result\_fastestLapSpeed <dbl> 218.300, 217.586,…  
## $ result\_finishDescription <chr> "Finished", "Fini…  
## $ result\_fastestLapTimeMillisec <int> 87500, 87700, 881…  
## $ status\_description <chr> "Finished", "Fini…  
## $ race\_round <int> 1, 1, 1, 1, 1, 1,…  
## $ race\_date <date> 2008-03-16, 2008…  
## $ driverStanding\_runningTotalPointsInSeason <int> 10, 8, 6, 5, 4, 3…  
## $ driverStanding\_runningPositionInSeason <int> 1, 2, 3, 4, 5, 6,…  
## $ driverStanding\_runningTotalWinsInSeason <int> 1, 0, 0, 0, 0, 0,…  
## $ driver\_name <chr> "hamilton", "heid…  
## $ driver\_dob <date> 1985-01-07, 1977…  
## $ driver\_homeCountry <chr> "United Kingdom",…  
## $ driver\_lat <dbl> 55.37805, 51.1656…  
## $ driver\_long <dbl> -3.435973, 10.451…  
## $ constructorStanding\_runningTotalPointsInSeason <int> 14, 8, 9, 5, 14, …  
## $ constructorStanding\_runningPositionInSeason <int> 1, 3, 2, 4, 1, 2,…  
## $ constructorStanding\_runningTotalWinsInSeason <int> 1, 0, 0, 0, 1, 0,…  
## $ constructor\_name <chr> "mclaren", "bmw\_s…  
## $ constructor\_homeCountry <chr> "United Kingdom",…  
## $ constructor\_lat <dbl> 55.37805, 51.1656…  
## $ constructor\_long <dbl> -3.435973, 10.451…  
## $ constructorResult\_pointsPerRace <int> 14, 8, 9, 5, 14, …  
## $ circuit\_name <chr> "albert\_park", "a…  
## $ circuit\_city <chr> "Melbourne", "Mel…  
## $ circuit\_country <chr> "Australia", "Aus…

summary(resultsHistorical)

## resultId raceId result\_startingGridPosition  
## Min. : 1 Min. : 1.0 Min. : 0.00   
## 1st Qu.: 6027 1st Qu.:276.8 1st Qu.: 5.00   
## Median :12038 Median :484.0 Median :11.00   
## Mean :11990 Mean :491.3 Mean :11.28   
## 3rd Qu.:17947 3rd Qu.:723.0 3rd Qu.:17.00   
## Max. :23781 Max. :988.0 Max. :34.00   
##   
## result\_finishOrder result\_pointsEarned result\_lapsCompleted  
## Min. : 1.00 Min. : 0.000 Min. : 0.0   
## 1st Qu.: 7.00 1st Qu.: 0.000 1st Qu.: 20.0   
## Median :13.00 Median : 0.000 Median : 52.0   
## Mean :13.08 Mean : 1.602 Mean : 45.3   
## 3rd Qu.:19.00 3rd Qu.: 1.000 3rd Qu.: 66.0   
## Max. :39.00 Max. :50.000 Max. :200.0   
##   
## result\_finishTimeMillisec result\_fastestLap result\_fastestLapRank  
## Min. : 1474899 Min. : 2.00 Min. : 0.0   
## 1st Qu.: 5444034 1st Qu.:30.00 1st Qu.: 5.0   
## Median : 5860280 Median :44.00 Median :11.0   
## Mean : 6300771 Mean :41.16 Mean :10.6   
## 3rd Qu.: 6495378 3rd Qu.:53.00 3rd Qu.:16.0   
## Max. :15090540 Max. :78.00 Max. :24.0   
## NA's :18004 NA's :18533 NA's :18378   
## result\_fastestLapSpeed result\_finishDescription  
## Min. : 89.54 Length:24104   
## 1st Qu.:190.53 Class :character   
## Median :202.31 Mode :character   
## Mean :200.68   
## 3rd Qu.:213.78   
## Max. :257.32   
## NA's :18534   
## result\_fastestLapTimeMillisec status\_description race\_round   
## Min. : 67400 Length:24104 Min. : 1.000   
## 1st Qu.: 81800 Class :character 1st Qu.: 4.000   
## Median : 91400 Mode :character Median : 8.000   
## Mean : 92140 Mean : 8.186   
## 3rd Qu.:100800 3rd Qu.:12.000   
## Max. :202300 Max. :21.000   
## NA's :18533   
## race\_date driverStanding\_runningTotalPointsInSeason  
## Min. :1950-05-13 Min. : 0.00   
## 1st Qu.:1975-06-22 1st Qu.: 0.00   
## Median :1989-05-07 Median : 3.00   
## Mean :1988-04-30 Mean : 14.09   
## 3rd Qu.:2003-06-18 3rd Qu.: 13.00   
## Max. :2017-11-26 Max. :397.00   
## NA's :482   
## driverStanding\_runningPositionInSeason  
## Min. : 1.00   
## 1st Qu.: 7.00   
## Median : 13.00   
## Mean : 15.02   
## 3rd Qu.: 21.00   
## Max. :105.00   
## NA's :482   
## driverStanding\_runningTotalWinsInSeason driver\_name   
## Min. : 0.0000 Length:24104   
## 1st Qu.: 0.0000 Class :character   
## Median : 0.0000 Mode :character   
## Mean : 0.3303   
## 3rd Qu.: 0.0000   
## Max. :13.0000   
## NA's :482   
## driver\_dob driver\_homeCountry driver\_lat   
## Min. :1896-12-28 Length:24104 Min. :-40.90   
## 1st Qu.:1943-11-21 Class :character 1st Qu.: 37.09   
## Median :1959-02-11 Mode :character Median : 46.23   
## Mean :1958-03-09 Mean : 36.46   
## 3rd Qu.:1974-07-13 3rd Qu.: 55.38   
## Max. :1998-10-29 Max. : 61.92   
##   
## driver\_long constructorStanding\_runningTotalPointsInSeason  
## Min. :-106.3468 Min. : 0.00   
## 1st Qu.: -3.4360 1st Qu.: 1.00   
## Median : 2.2138 Median : 9.00   
## Mean : 0.6719 Mean : 30.52   
## 3rd Qu.: 12.5674 3rd Qu.: 31.00   
## Max. : 174.8860 Max. :765.00   
## NA's :1876   
## constructorStanding\_runningPositionInSeason  
## Min. : 1.000   
## 1st Qu.: 3.000   
## Median : 6.000   
## Mean : 6.566   
## 3rd Qu.: 9.000   
## Max. :21.000   
## NA's :1876   
## constructorStanding\_runningTotalWinsInSeason constructor\_name   
## Min. : 0.0000 Length:24104   
## 1st Qu.: 0.0000 Class :character   
## Median : 0.0000 Mode :character   
## Mean : 0.7522   
## 3rd Qu.: 1.0000   
## Max. :19.0000   
## NA's :1876   
## constructor\_homeCountry constructor\_lat constructor\_long   
## Length:24104 Min. :-40.90 Min. :-106.347   
## Class :character 1st Qu.: 41.87 1st Qu.: -3.436   
## Mode :character Median : 53.41 Median : -3.436   
## Mean : 48.30 Mean : 5.215   
## 3rd Qu.: 55.38 3rd Qu.: 12.567   
## Max. : 61.52 Max. : 174.886   
##   
## constructorResult\_pointsPerRace circuit\_name circuit\_city   
## Min. : 0.000 Length:24104 Length:24104   
## 1st Qu.: 0.000 Class :character Class :character   
## Median : 0.000 Mode :character Mode :character   
## Mean : 3.408   
## 3rd Qu.: 4.000   
## Max. :66.000   
## NA's :1572   
## circuit\_country   
## Length:24104   
## Class :character   
## Mode :character   
##   
##   
##   
##

colSums(is.na(resultsHistorical))

## resultId   
## 0   
## raceId   
## 0   
## result\_startingGridPosition   
## 0   
## result\_finishOrder   
## 0   
## result\_pointsEarned   
## 0   
## result\_lapsCompleted   
## 0   
## result\_finishTimeMillisec   
## 18004   
## result\_fastestLap   
## 18533   
## result\_fastestLapRank   
## 18378   
## result\_fastestLapSpeed   
## 18534   
## result\_finishDescription   
## 0   
## result\_fastestLapTimeMillisec   
## 18533   
## status\_description   
## 0   
## race\_round   
## 0   
## race\_date   
## 0   
## driverStanding\_runningTotalPointsInSeason   
## 482   
## driverStanding\_runningPositionInSeason   
## 482   
## driverStanding\_runningTotalWinsInSeason   
## 482   
## driver\_name   
## 0   
## driver\_dob   
## 0   
## driver\_homeCountry   
## 0   
## driver\_lat   
## 0   
## driver\_long   
## 0   
## constructorStanding\_runningTotalPointsInSeason   
## 1876   
## constructorStanding\_runningPositionInSeason   
## 1876   
## constructorStanding\_runningTotalWinsInSeason   
## 1876   
## constructor\_name   
## 0   
## constructor\_homeCountry   
## 0   
## constructor\_lat   
## 0   
## constructor\_long   
## 0   
## constructorResult\_pointsPerRace   
## 1572   
## circuit\_name   
## 0   
## circuit\_city   
## 0   
## circuit\_country   
## 0

head(resultsHistorical)

## resultId raceId result\_startingGridPosition result\_finishOrder  
## 1 1 18 1 1  
## 2 2 18 5 2  
## 3 3 18 7 3  
## 4 4 18 11 4  
## 5 5 18 3 5  
## 6 6 18 13 6  
## result\_pointsEarned result\_lapsCompleted result\_finishTimeMillisec  
## 1 10 58 5690616  
## 2 8 58 5696094  
## 3 6 58 5698779  
## 4 5 58 5707797  
## 5 4 58 5708630  
## 6 3 57 NA  
## result\_fastestLap result\_fastestLapRank result\_fastestLapSpeed  
## 1 39 2 218.300  
## 2 41 3 217.586  
## 3 41 5 216.719  
## 4 58 7 215.464  
## 5 43 1 218.385  
## 6 50 14 212.974  
## result\_finishDescription result\_fastestLapTimeMillisec  
## 1 Finished 87500  
## 2 Finished 87700  
## 3 Finished 88100  
## 4 Finished 88600  
## 5 Finished 87400  
## 6 Finished 89600  
## status\_description race\_round race\_date  
## 1 Finished 1 2008-03-16  
## 2 Finished 1 2008-03-16  
## 3 Finished 1 2008-03-16  
## 4 Finished 1 2008-03-16  
## 5 Finished 1 2008-03-16  
## 6 +1 Lap 1 2008-03-16  
## driverStanding\_runningTotalPointsInSeason  
## 1 10  
## 2 8  
## 3 6  
## 4 5  
## 5 4  
## 6 3  
## driverStanding\_runningPositionInSeason  
## 1 1  
## 2 2  
## 3 3  
## 4 4  
## 5 5  
## 6 6  
## driverStanding\_runningTotalWinsInSeason driver\_name driver\_dob  
## 1 1 hamilton 1985-01-07  
## 2 0 heidfeld 1977-05-10  
## 3 0 rosberg 1985-06-27  
## 4 0 alonso 1981-07-29  
## 5 0 kovalainen 1981-10-19  
## 6 0 nakajima 1985-01-11  
## driver\_homeCountry driver\_lat driver\_long  
## 1 United Kingdom 55.37805 -3.435973  
## 2 Germany 51.16569 10.451526  
## 3 Germany 51.16569 10.451526  
## 4 Spain 40.46367 -3.749220  
## 5 Finland 61.92411 25.748151  
## 6 Japan 36.20482 138.252924  
## constructorStanding\_runningTotalPointsInSeason  
## 1 14  
## 2 8  
## 3 9  
## 4 5  
## 5 14  
## 6 9  
## constructorStanding\_runningPositionInSeason  
## 1 1  
## 2 3  
## 3 2  
## 4 4  
## 5 1  
## 6 2  
## constructorStanding\_runningTotalWinsInSeason constructor\_name  
## 1 1 mclaren  
## 2 0 bmw\_sauber  
## 3 0 williams  
## 4 0 renault  
## 5 1 mclaren  
## 6 0 williams  
## constructor\_homeCountry constructor\_lat constructor\_long  
## 1 United Kingdom 55.37805 -3.435973  
## 2 Germany 51.16569 10.451526  
## 3 United Kingdom 55.37805 -3.435973  
## 4 France 46.22764 2.213749  
## 5 United Kingdom 55.37805 -3.435973  
## 6 United Kingdom 55.37805 -3.435973  
## constructorResult\_pointsPerRace circuit\_name circuit\_city  
## 1 14 albert\_park Melbourne  
## 2 8 albert\_park Melbourne  
## 3 9 albert\_park Melbourne  
## 4 5 albert\_park Melbourne  
## 5 14 albert\_park Melbourne  
## 6 9 albert\_park Melbourne  
## circuit\_country  
## 1 Australia  
## 2 Australia  
## 3 Australia  
## 4 Australia  
## 5 Australia  
## 6 Australia

#normalize laps completed field  
resultsHistorical <- resultsHistorical %>%  
 group\_by(raceId) %>%  
 mutate(maxLaps = max(result\_lapsCompleted),  
 result\_percentOfRaceCompleted = (result\_lapsCompleted / maxLaps) \* 100,  
 result\_lapsCompleted = NULL,  
 maxLaps = NULL  
 )  
  
# ungroup columns  
resultsHistorical <- resultsHistorical %>%  
 ungroup() %>%  
 as.data.frame()  
  
# remove resultId column which is no longer needed  
resultsHistorical <- subset(resultsHistorical, select = -resultId)  
  
  
# remove result\_finishTimeMillisec due to difficult to handle NAs  
resultsHistorical <- subset(resultsHistorical, select = -result\_finishTimeMillisec)  
  
  
# remove columns relating to fastest lap data which contain many untreatable NAs  
resultsHistorical <- subset(resultsHistorical, select = -c(result\_fastestLap:result\_fastestLapSpeed, result\_fastestLapTimeMillisec))  
  
  
# fill NAs in 'points' & 'wins' related columns with 0 and remove 'position' related columns  
 # NAs in 'driverStandings.' and 'consturctorStandings.' caused by failures to qualify in early season races  
 # attempt later -- [fill NAs in 'position' columns with \*max runningPositionsInSeason + 1\* for each season]  
 # use grouped mutate (window function)  
resultsHistorical$driverStanding\_runningTotalPointsInSeason[is.na(resultsHistorical$driverStanding\_runningTotalPointsInSeason)] <- 0  
  
resultsHistorical$driverStanding\_runningTotalWinsInSeason[is.na(resultsHistorical$driverStanding\_runningTotalWinsInSeason)] <- 0  
  
resultsHistorical <- subset(resultsHistorical, select = -driverStanding\_runningPositionInSeason)  
  
resultsHistorical$constructorStanding\_runningTotalPointsInSeason[is.na(resultsHistorical$constructorStanding\_runningTotalPointsInSeason)] <- 0  
  
resultsHistorical$constructorStanding\_runningTotalWinsInSeason[is.na(resultsHistorical$constructorStanding\_runningTotalWinsInSeason)] <- 0  
  
resultsHistorical$constructorResult\_pointsPerRace[is.na(resultsHistorical$constructorResult\_pointsPerRace)] <- 0  
  
resultsHistorical <- subset(resultsHistorical, select = -constructorStanding\_runningPositionInSeason)  
  
  
# create driver age at time of race feature and convert to int  
# remove 'driver\_dob' (consider re-adding in future to analyze race results on closeness to birthday)  
resultsHistorical <- resultsHistorical %>%  
 mutate(driverAge = (race\_date - driver\_dob) / 365,  
 driver\_dob = NULL  
 )  
  
resultsHistorical$driverAge <- as.integer(resultsHistorical$driverAge)  
  
# correct outlier  
resultsHistorical[17012, "driverAge"] = 33  
  
  
# convert race date to month and year columns (for better factorization)  
# remove 'race\_date'  
resultsHistorical <- resultsHistorical %>%  
 mutate(race\_month = month(race\_date),  
 race\_year = year(race\_date),  
 race\_date = NULL  
 )  
  
  
# create race in driver home country? and constructor home country (True or False) feature and remove existing fields  
# resultsHistorical <- resultsHistorical %>%  
 # mutate(driverHomeCountry = (substr(circuit\_country, 1, 3) == substr(driver\_nationality, 1, 3)),  
 # constructorHomeCountry = (substr(circuit\_country, 1, 3) == substr(constructor\_nationality, 1, 3)),  
 # driver\_nationality = NULL,  
 # constructor\_nationality = NULL,  
 # circuit\_country = NULL  
 # )  
  
  
# create running total points going into race for driver and constructor and remove existing columns  
resultsHistorical <- resultsHistorical %>%  
 mutate(driver\_preRaceTotPoints = driverStanding\_runningTotalPointsInSeason - result\_pointsEarned,  
 constructor\_preRaceTotPoints = constructorStanding\_runningTotalPointsInSeason - constructorResult\_pointsPerRace,  
 )  
  
  
# correct for inaccurate observations (i.e. negative points) caused by NAs from disqualification, failure to qualify, etc.  
# 1 error in driverPoints  
resultsHistorical$driver\_preRaceTotPoints <- ifelse(resultsHistorical$driver\_preRaceTotPoints < 0, 0, resultsHistorical$driver\_preRaceTotPoints)  
  
# ~50 errors in constructor points  
resultsHistorical$constructor\_preRaceTotPoints <- ifelse(resultsHistorical$constructor\_preRaceTotPoints < 0, 0, resultsHistorical$constructor\_preRaceTotPoints)  
  
  
# remove unnecessary columns  
resultsHistorical <- subset(resultsHistorical, select = -c(driverStanding\_runningTotalPointsInSeason, constructorStanding\_runningTotalPointsInSeason))  
  
  
# create running total wins going into race for driver and constructor and remove existing columns  
resultsHistorical <- resultsHistorical %>%  
 mutate(driver\_preRaceTotWins = ifelse(result\_finishOrder == 1, driverStanding\_runningTotalWinsInSeason - 1, driverStanding\_runningTotalWinsInSeason),  
 driverStanding\_runningTotalWinsInSeason = NULL  
 )  
  
resultsHistorical <- resultsHistorical %>%  
 group\_by(raceId, constructor\_name) %>%  
 mutate(constructor\_preRaceTotWins = ifelse(min(result\_finishOrder) == 1, constructorStanding\_runningTotalWinsInSeason - 1, constructorStanding\_runningTotalWinsInSeason),  
 constructorStanding\_runningTotalWinsInSeason = NULL  
 )  
  
  
# ungroup columns  
resultsHistorical <- resultsHistorical %>%  
 ungroup() %>%  
 as.data.frame()  
  
  
# correct for inaccurate observations (i.e. negative points) caused by NAs from disqualification, failure to qualify, etc.  
# >200 in constructorTotWins  
resultsHistorical$constructor\_preRaceTotWins <- ifelse(resultsHistorical$constructor\_preRaceTotWins < 0, 0, resultsHistorical$constructor\_preRaceTotWins)  
  
  
# remove unnecessary columns  
resultsHistorical <- subset(resultsHistorical, select = -c(raceId, constructorResult\_pointsPerRace))  
  
  
# create features which contain previous race results and remove current race result fields  
resultsHistorical <- resultsHistorical %>%  
 arrange(driver\_name, race\_year, race\_round) %>%  
 mutate(result\_percentOfPreviousRaceCompleted = ifelse(driver\_name == lag(driver\_name) & !is.na(lag(driver\_name)), lag(result\_percentOfRaceCompleted), 0),  
 result\_previousFinishDescrip = ifelse(driver\_name == lag(driver\_name) & !is.na(lag(driver\_name)), lag(result\_finishDescription), "First race"),  
 status\_previousDescrip = ifelse(driver\_name == lag(driver\_name) & !is.na(lag(driver\_name)), lag(status\_description), "First race"),  
 result\_percentOfRaceCompleted = NULL,  
 result\_finishDescription = NULL,  
 status\_description = NULL  
 )  
  
  
# convert columns (driverAge, percentOfPreviousRaceCompleted, preRaceTotPoints, preRaceTotWins) to integers  
resultsHistorical$driverAge <- as.integer(resultsHistorical$driverAge)  
resultsHistorical$result\_percentOfPreviousRaceCompleted <- as.integer(resultsHistorical$result\_percentOfPreviousRaceCompleted)  
resultsHistorical$driver\_preRaceTotPoints <- as.integer(resultsHistorical$driver\_preRaceTotPoints)  
resultsHistorical$driver\_preRaceTotWins <- as.integer(resultsHistorical$driver\_preRaceTotWins)  
  
  
# convert pointsEarned to boolean  
resultsHistorical <- resultsHistorical %>%  
 mutate(result\_inThePoints = ifelse(result\_pointsEarned > 0, TRUE, FALSE),  
 result\_pointsEarned = NULL  
 )  
  
  
# arrange rows  
resultsHistorical <- resultsHistorical %>%  
 arrange(race\_year, race\_round, result\_finishOrder, result\_startingGridPosition)  
  
  
# arrange columns  
resultsHistorical <- resultsHistorical[, c("result\_finishOrder", "result\_inThePoints", "result\_startingGridPosition",   
 "driver\_name", "constructor\_name", "circuit\_name", "circuit\_city", "race\_month",  
 "race\_year", "race\_round", "driverAge", "driver\_homeCountry", "driver\_lat", "driver\_long",  
 "constructor\_homeCountry", "constructor\_lat", "constructor\_long",  
 "result\_percentOfPreviousRaceCompleted", "result\_previousFinishDescrip",  
 "status\_previousDescrip", "driver\_preRaceTotPoints", "constructor\_preRaceTotPoints",  
 "driver\_preRaceTotWins", "constructor\_preRaceTotWins")]  
  
  
# remove columns due to appeared unreliability  
resultsHistorical <- subset(resultsHistorical, select = -c(constructor\_preRaceTotPoints, constructor\_preRaceTotWins))  
  
  
  
  
#### Write file ####  
  
# define output path  
path\_out2 <- "../Analyzed Data"  
  
# write to csv  
write.csv(resultsHistorical, file.path(path\_out2, "resultsHistorical.csv"), row.names = F)