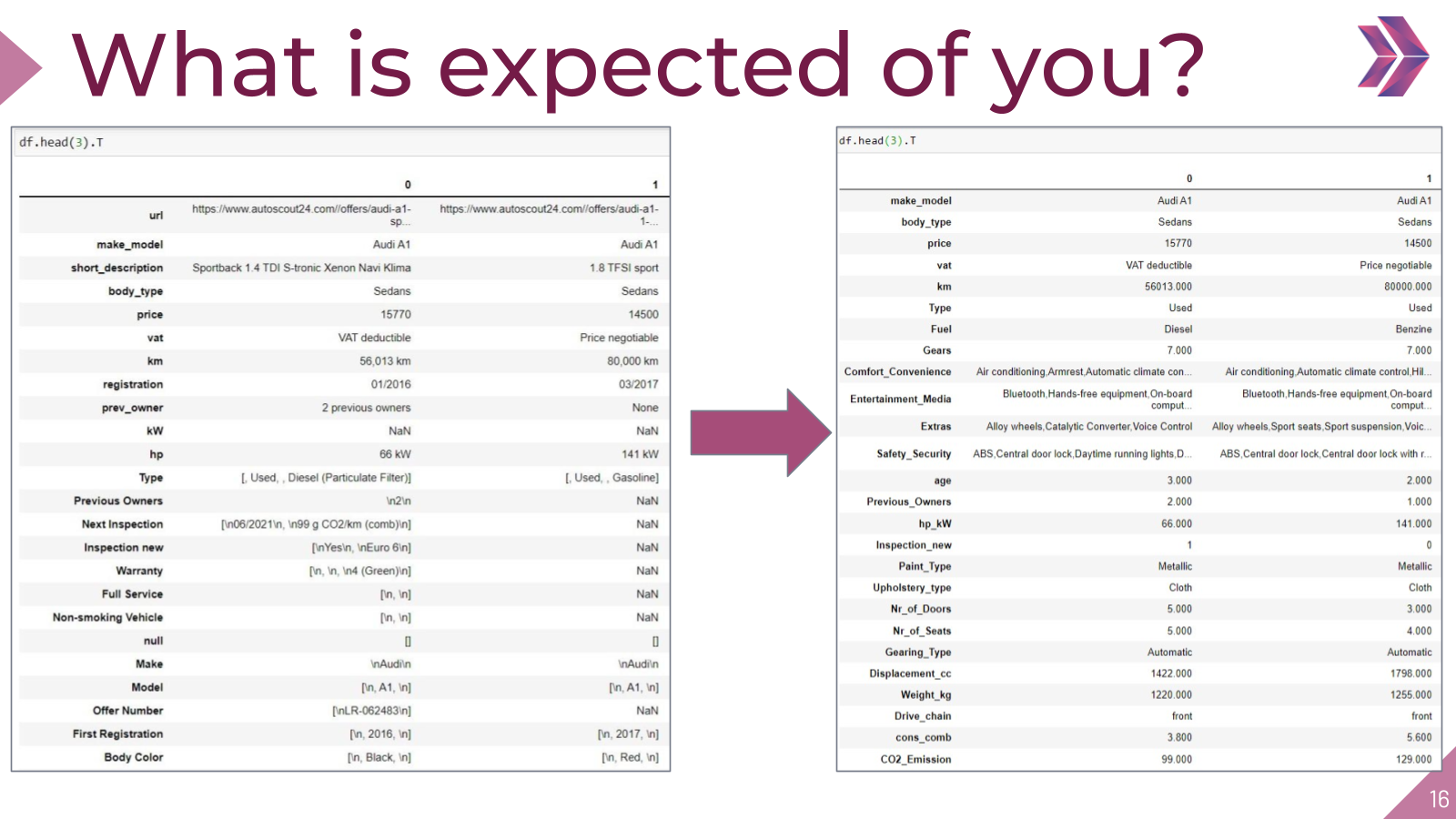
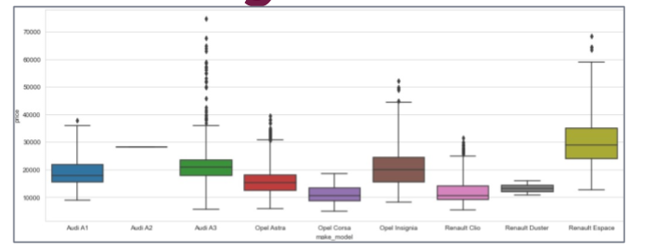
* **AMAÇ:** Daha sonra Machine Learning model de girdi (input) olarak kullanmak üzere araba fiyatlarının tahmin edilmesi ve bunun EDA (Exploratory Data Analysis) ile yapılması
* .json dosyası **15919** satır ve **54** sütundan oluşuyor. Online araba ticareti yapan bir firmanın 2019 verilerinden alınmış. **9** farklı araba modeliyle ilgili veriler ve özellikler içeriyor.
* Bu veri setinin özellikleri(değişkenleri) çok dağınık ve bozulmuş durumda.
* **BEKLENENLER:** 
  + **Bozuk veri formatlarını** düzenleyin
  + Alan bilgisi(**domain knowledge**, burada araba modelleri ve piyasası) önemli
  + Bir verinin **outlier olup olmadığına karar verirken** dikkatli olun, **her outlier atılmalı diye bir şey yok**
  + Analiz sonucunda gereksiz olduğunu düşündüğünüz sütun/satırları silin
  + Bu süreçler boyunca görselleştirme araçlarını kullanın



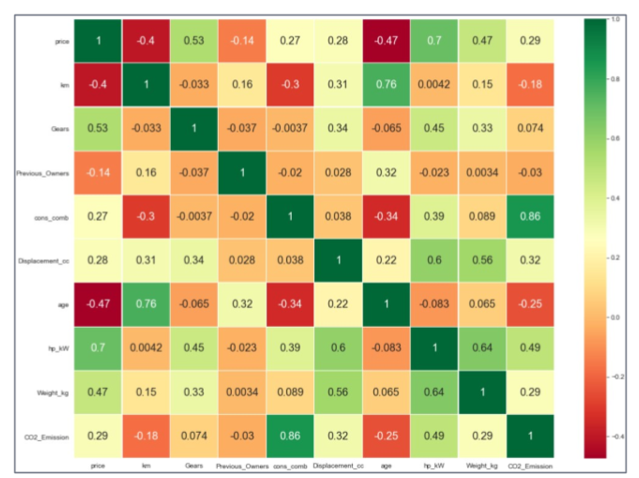
Yukarıdaki örnekte df.head(3).T yapılarak nasıl bir düzeltme yapılacağı gösterilmiş:

* **url** sütunu tamamen kaldırılmış
* **short\_description** sütunu kaldırılmış
* **km** sütunu içinde sadece sayı bırakılmış, km ifadesi kaldırılmış
* **registration** sütunu kaldırılmış
* **prev\_owner** sütunu kaldırılmış
* **kW** sütunu kaldırılmış, zaten verilen örnekte NAN ifadeler içerdiği görülüyor
* **hp** sütunu kaldırılmış
* **Type** sütununda liste içinde Used, Diesedl(Particular Filter), Gasoline gibi ifadeler ve fazladan virgüller var, bunlar örneğin Used olarak ya da sadece gasoline olarak kalacak şekilde temizlenmiş. Yine burda hangi araba gasoline kullanıyor, hangisi used bilmek ve araştırmak lazım
* **Fuel ve Gears** diye iki sütun eklenmiş, bunlardaki bilgiler Type sütunundan alınmış olabilir çünkü Type sütununda hücrelerde liste şeklinde birçok veri bulunuyor
* **Comfort\_Convenience**, **Entertainment\_Media**, **Extras**, **Safety\_Security** ve **age** isimlerinde sütunlar oluşturulmuş. Bunlar ya silinen, ya düzenlenen sütunlardan alınan verilerle ya da internet araştırılmasıyla doldurulmuş
* **Previous Owners** sütununda \n2\n gibi ifadeler var, bunlar sayısal ifadelerle değiştirilmiş
* **hp\_kW** diye yeni bir sütun oluşturulmuş, muhtemelen yukarıda silinen kW ve hp sütunlarından alındı veriler ya da internet araştırmasından
* **Next Inspection** sütunu kaldırılmış, muhtemel içindeki verilerle yeni sütunlar oluşturuldu
* **Inspection new** sütununun ismi **Inspection\_new** olarak değiştirilmiş. İçeriğindeki liste içinde görülen anlamsız ifadeler de 1 ve 0 olarak değiştirilmiş gibi duruyor, yani bolean ifade, araba yeni ya da değil
* **Warranty** sütunu kaldırılmış
* **Full Service** sütunu kaldırılmış
* **Non-smoking Vehicle** sütunu kaldırılmış
* **null** sütunu kaldırılmış
* **Model** sütunu kaldırılmış
* **Offer Number** sütunu kaldırılmış
* **First Registration** sütunu kaldırılmış
* **Body Color** sütunu kaldırılmış
* **Paint\_Type** diye bir sütun oluşturulmuş, string ifadeler içeriyor, “Metalic” gibi
* **Upholstery\_type** diye bir sütun oluşturulmuş, string ifadeler içeriyor, “Cloth” gibi
* **Nr\_of\_Doors** diye bir sütun oluşturmuş, tam sayı ifadeler içeriyor
* **Nr\_of\_Seats** diye bir sütun oluşturulmuş, tam sayı ifadeler içeriyor
* **Gearing\_Type** diye bir sütun oluşturulmuş, string ifadeler içeriyor, “Automatic” gibi
* **Displacement\_cc** diye bir sütun oluşturulmuş, sayısal ifadeler içeriyor
* **Weight\_kg** diye bir sütun oluşturulmuş, sayısal ifadeler içeriyor
* **Drive\_chain** diye bir sütun oluşturulmuş, string ifadeler içeriyo, “front” gibi
* **Cons\_comb** diye bir sütun oluşturulmuş, sayısal ifadeler içeriyor
* **CO2\_Emission** diye bir sütun oluşturulmuş, sayısal ifadeler içeriyor

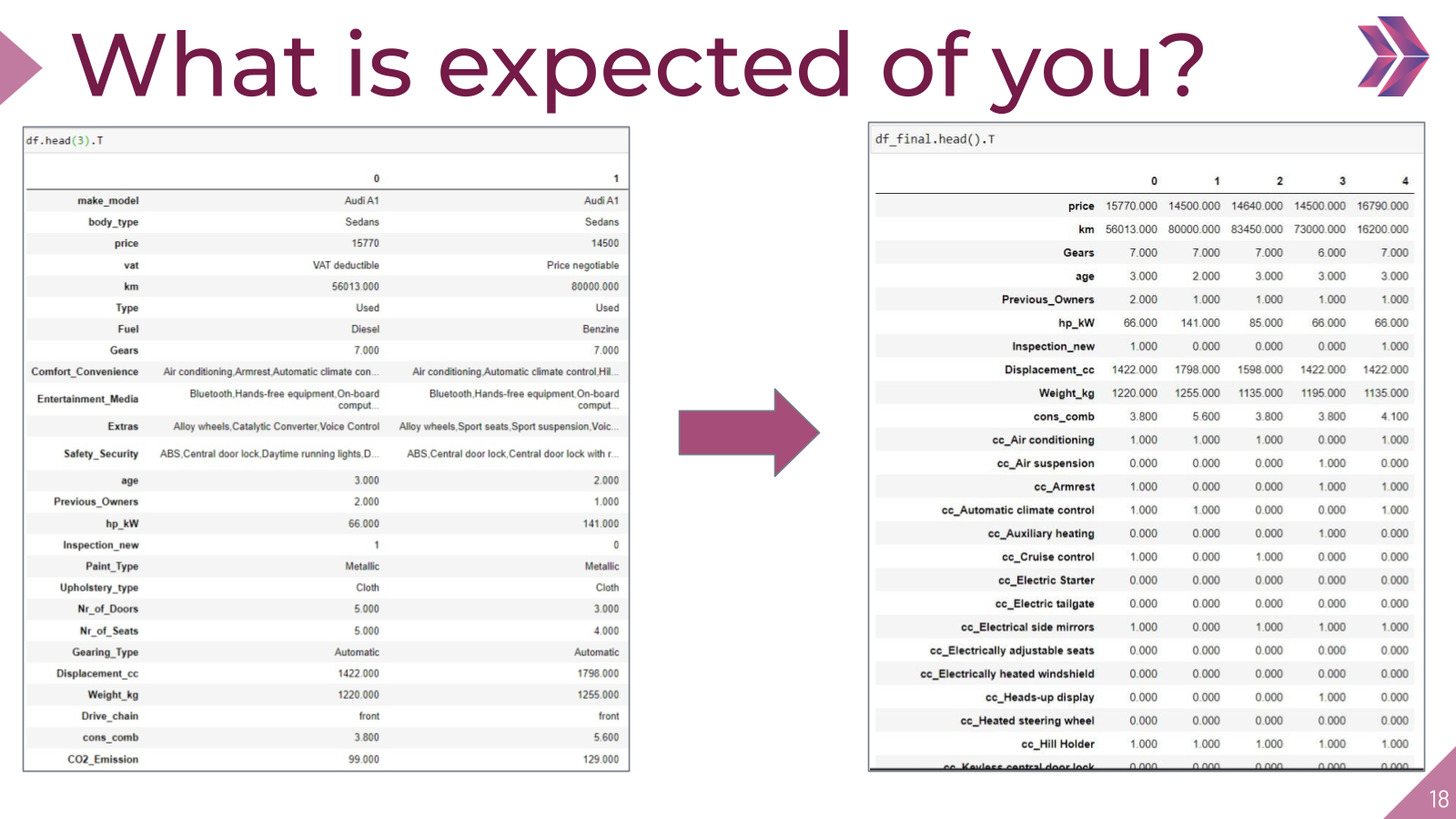
**Yukarıdaki yeni sütun ve verileri ya silinen sütunlardan ya da internet araştırmasıyla oluşturulmuş**



**Yukarıdaki resimde görüleceği üzere bir boxplot görselleştirmesi her bir araç modeli için yapılıp (make\_model sütunu kullanılmış sanırım) outlierlar belirlenmiş. Burada önemli olan her outlier atılmak zorunda değil, hoca bunun üzerinde durdu**

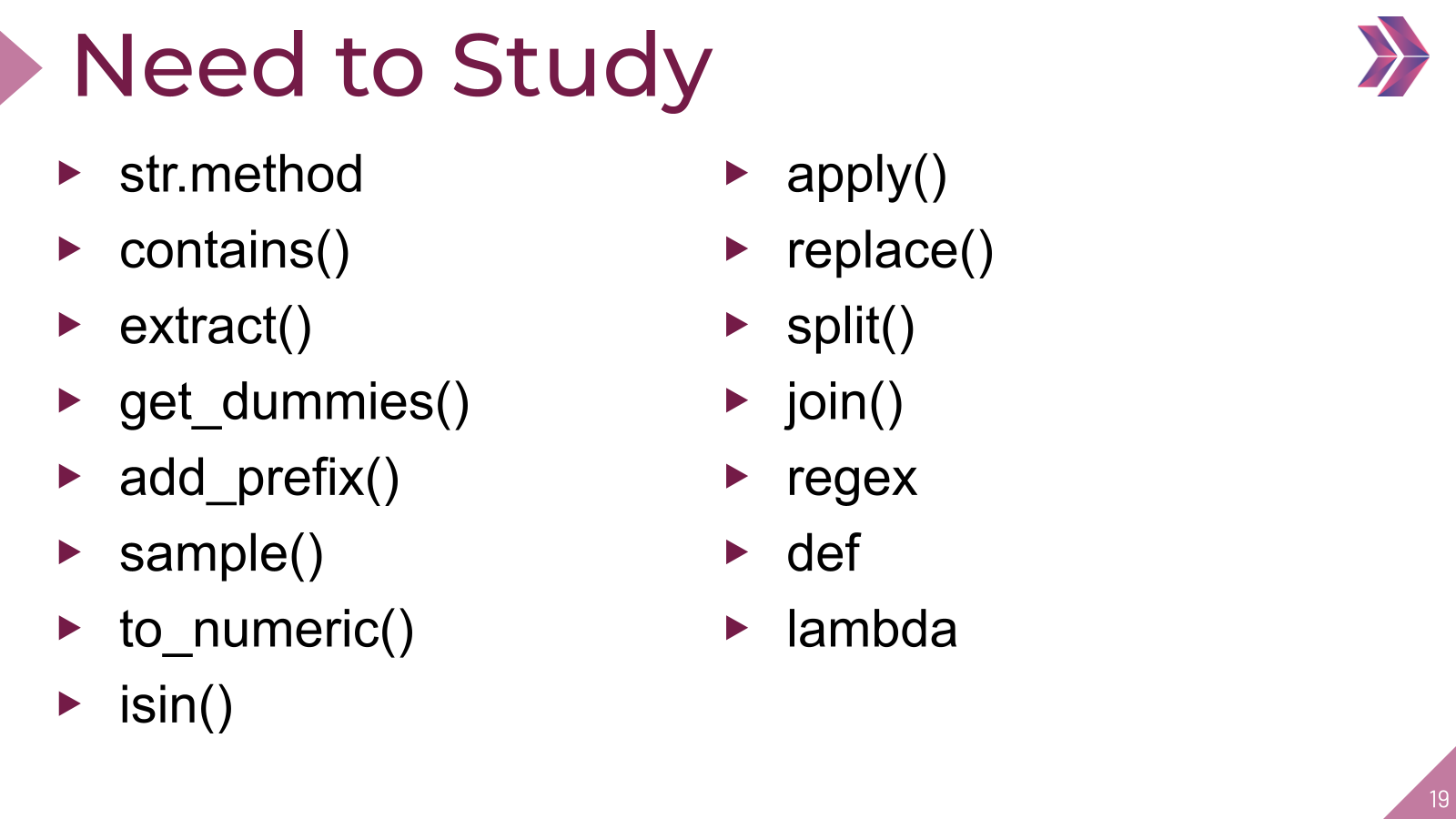
****

**Yine yukarıdaki resimde görüleceği üzere tüm sütunlar kullanılarak bir heatmap oluşturulmuş, covariance hesaplaması gerekiyor olabilir**



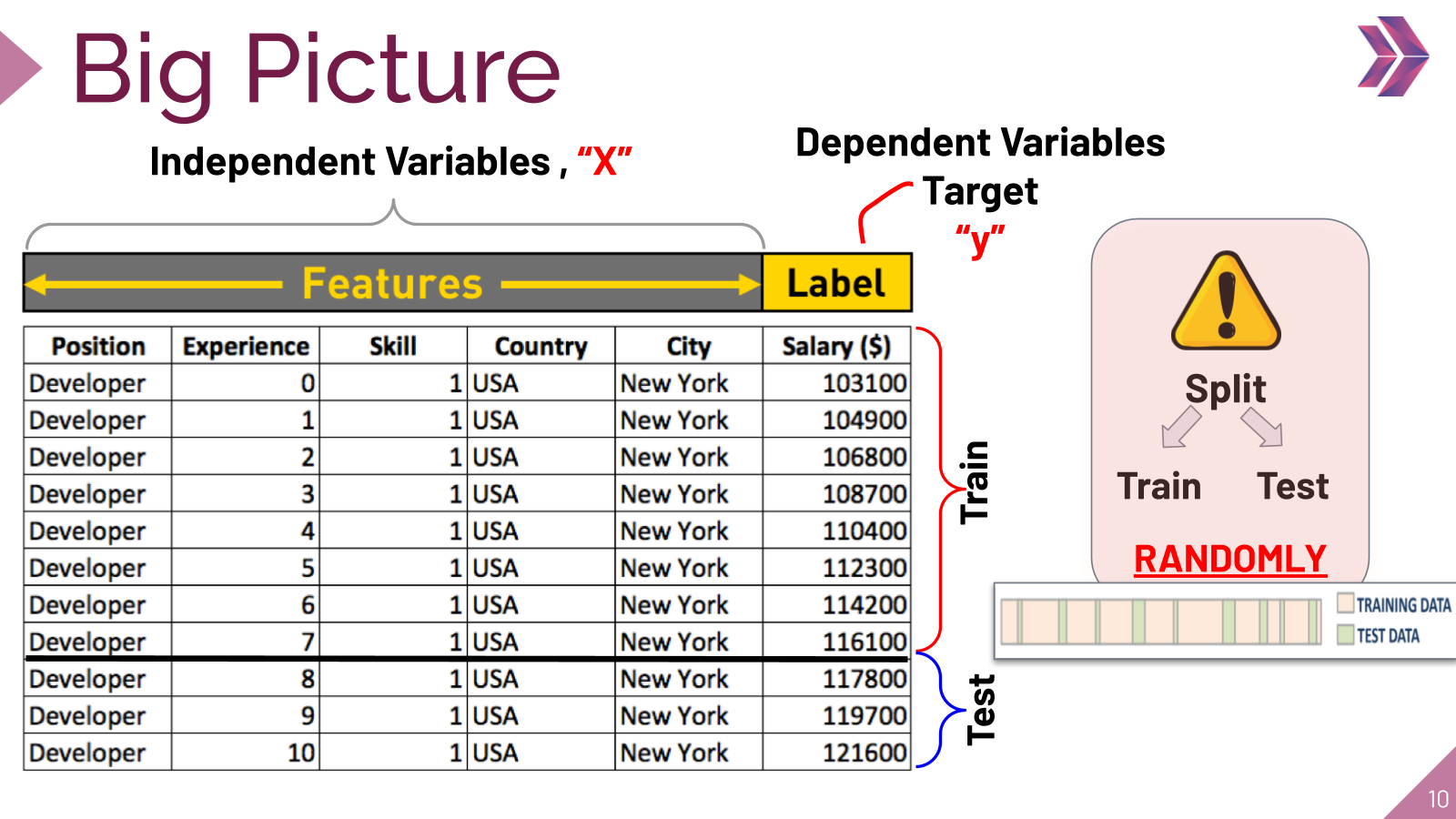
**Nihai aşamada temizlenmiş data seti Machine Learning modeline girebilmesi için sayısallaştırılacak(get\_dummy vs gibi metodlar) kullanılarak**

**İHTİYAÇ DUYULAN BİLGİLER**



**extract() metodu re modülünün içinde, Salih Mentörün lab notebookunda güzel anlatılmış**

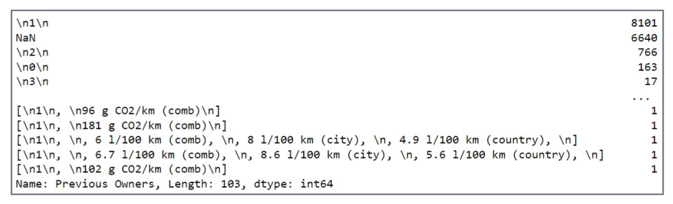
**NOTE: Şu anki yılın 2019 olduğu varsayılacak**



**İPUÇLARI:**

* **Domain knowledge çok önemli**
* **Her sütun hedef etikete (target label) göre değerlendirilmeli**
* **Sütun isimleri kontrol edilecek**
* **Her bir sütundaki null değerlerin yüzdeleri kontrol edilecek, belli bir oranı geçen sütunlar atılabilinir**
* **Her bir sütun için value\_counts() yapılacak, bunlar değerlendirilip ne yapılacağına dair notlar alınacak (atılacaklar, sütunlar arasındaki benzerlikler, nasıl temizlemeli, şablonu tanımla gibi)**

**HER BİR SÜTUNDAKİ LİSTEDEN HER BİR DEĞER NASIL ÇIKARTILIR?**

****

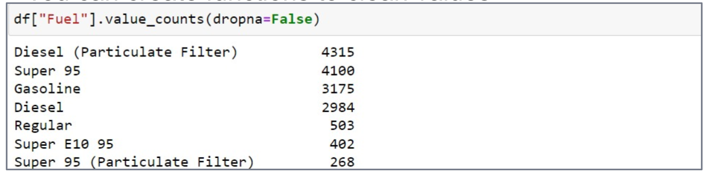
df["Previous\_Owners"] = [item[0] if type(item) == list else item for item in df["Previous Owners"]]

df["Previous\_Owners"]

df["Previous\_Owners2"] = df["Previous Owners"].apply(lambda item: item[0] if type(item) == list else item for item)

df["Previous\_Owners2"]

**VERİLERİ TEMİZLEMEK İÇİN FONKSİYONLAR OLUŞTURABİLİRSİNİZ**

****

benzine = ["Gasoline", "Super 95", "Regular", "Super E10 95", "Super Plus 98", "Super Plus E10 98", "Others"]

lpg = ["LPG", "Liquid petroleum gas", "CNG", "Biogas", "Domestic gas H"]

def fueltype(x)

if x in benzine:

return "Benzine"

elif x in lpg:

return "LPG/CNG"

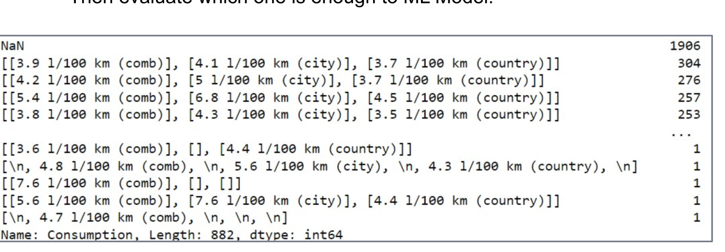
else:

return x

df["Fuel"] = df.Feuel.apply(fueltype)

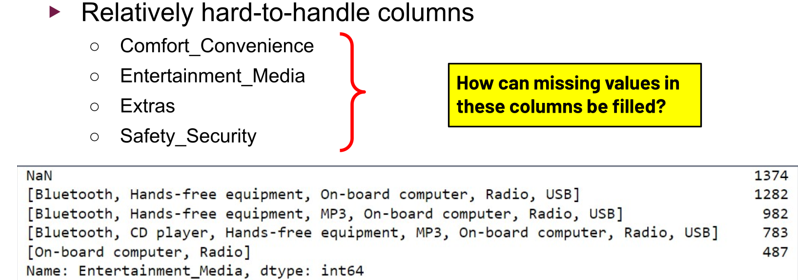
**GÖRECELİ DÜZELTMESİ ZOR SÜTUNLAR**

**TÜKETİM**

****

Ayrı sütunlar oluşturmak için he bir tüketim tipi için şablonlar tanımla. Daha sonra hangisinin ML Modeli için olacağına karar ver

**GÖRECELİ DÜZELTMESİ ZOR SÜTUNLAR**



Comfort\_Convenience, Entertainment\_Media, Extras ve Safety\_Security gibi yeni sütunlardaki verileri nasıl dolduracağız?

df["Entertainment\_Media"] = [",.join(item) if type(item) == list else item for item in df[“Entertainment\_Media"]]

**EKSİK VERİLERİ TAMAMLAMAK İÇİN SÜTUNLARI NASIL İNCELEYECEĞİZ?**

df.groupby("age").km.describe()

df.groupby(["make\_model", "age"]).km.describe()

df.groupby(["make\_model", "body\_type", "age"]).price.describe()

**KAYIP VERİLERİ GRUPLAYARAK NASIL TAMAMLARIZ – ÖRNEK 1**

**Birinci aşama:**

df["body\_type"].fillna(df["body\_type"].mode()[0])

**İkinci aşama:**

df.loc[df["make\_model"] == "Audi A1", "body\_type"].fillna(df[df["make\_model"]=="Audi A1"]["body\_type"].mode()[0])

**Üçüncü aşama: (Kendi fonksiyonunuzu oluşturmak için bu loop’u genelleştirebilirsiniz)**

for group in list((df["make\_model"]).unique()):

cond = df["make\_model"]==group

mode = list(df[cond]["body\_type"].mode())

if mode != []:

df.loc[cond, "body\_type"] = df.loc[cond, "body\_type"].fillna(df[cond]["body\_type"].mode()[0])

else:

df.loc[cond, "body\_type"] = df.loc[cond, "body\_type"].fillna(df["body\_type"].mode()[0])

**KAYIP VERİLERİ GRUPLAYARAK NASIL TAMAMLARIZ – ÖRNEK 2**

**Birinci aşama:**

df.["Previous\_Owners"].fillna(method="ffill")

**İkinci aşama:**

df.loc[df["age"]==0, "Previous\_Owners"].fillna(method="ffill")

**Üçüncü aşama: (Kendi fonksiyonunuzu oluşturmak için bu loop’u genelleştirebilirsiniz)**

for group in list(df["age"].unique()):

cond = df["age"] == group

df.loc[cond, "Previous\_Owners"] = df.loc[cond, "Previous\_Owners]"].fillna(method="ffill").fillna(method="bfill")

df["Previous\_Owners"] = df["Previous\_Owners"].fillna(method="ffill").fillna(method="bfill")

**KAYIP VERİLERİ GRUPLAYARAK NASIL TAMAMLARIZ – ÖRNEK 3**

**Birinci aşama:**

df.["Paint\_Type"].fillna(method="ffill")

**İkinci aşama:**

df.loc[df["age"]==0, "Paint\_Type"].fillna(method="ffill")

**Üçüncü aşama:**

for group in list(df["make\_model"].unique()):

cond = df["make\_model"]==group

df.loc[cond, "Point\_Type"] = df.loc[cond, "Paint\_Type"].fillna(method="ffill").fillna(method="bfill")

df["Paint\_Type"] = df["Paint\_Type"].fillna(method="ffill").fillna(method="bfill")

**DUMMY İŞLEMLERİ**

get\_dummies fonksiyonu iki türlü kullanılır.

pd.get\_dummies(df)

df[“column\_name”].str.get\_dummies(sep=”,”)