1.ImageDataBunch ile indirilen Google images'dan bir dataset yaratırken:

- Dosyaların path'ini veriyoruz, train='.' dediğimizde current folderı training set olarak kabul et demiş oluyoruz. Normalde, train adında bir klasör arayacaktı.
- Bunun yanında current klasördeki dataların %20'sini ayır diyoruz.
- Ayrıca validation set'in her zaman aynı olması için random.seed ekiyoruz yoksa bir hyperparameter change denediğimizde validation set farklı olursa, change'in işe yarayıp yaramadığından emin olamayız. Belki yeni seçilen validation set daha kolay veya zor olabilir.

2.Data cleaning sırasında .top_losses örneklere bakıyoruz:

- Bu örnekler, modelin en çok yanıldığı veya en az emin oldukları olacak.
- Eğer mislabeled data varsa, modelin bunu yüksek confidence ile doğru classify etmesi çok zor. Bu yüzden muhtemelen top_losses bize bu örnekleri gösterecek.
- Yani cleaning sırasında gördüğümüz örnekler, ya modelin yanlış tahmin ettiği örnekler olacak ya da doğru tahmin ettiği ancak emin olmadığı örnekler olacak!

3.%90 Rule of Thumbs kullanırsak, modelimiz çok iyi çalışır, ancak bazen sıkıntı çıkabilir.4 temel sıkıntıdan bahsedebiliriz. Daha başkaları da var ama en önemli 4'ü bu.

- Sıkıntının sebebi learning rate'in çok büyük veya küçük olması olabilir.
- Ya da epoch sayısı olabilir.
- Eğer learning rate çok büyük olursa, validation loss uçacaktır. Optimizasyon algoritması minimum'a converge edemeyeceği için bu normal, bence training loss da uçmalı zaten o da artmış ama validation kadar değil, neden bilmiyorum. Belki de learning rate'i daha da büyütsek o da uçar. Bunu düzeltmek için modeli baştan yaratmalıyız ve fit sırasında learning rate'i daha küçük seçmeliyiz.
- Learning rate çok küçük olursa, her adımda error rate daha iyi olur, ancak bu işlem çok yavaş olur. Yani gradient step çok küçüktür, convergence çok uzun sürecek.
- Eğer lr çok küçükse göreceğimiz bir başka olgu: **training loss > valid loss**'tur böyle bir modeli hiçbir zaman istemeyiz. Bu durum her zaman we haven't fitted enough'ın göstergesidir.
- Bu durumda ya learning rate'imiz çok küçüktür veya number of epochs çok küçüktür. Böyle bir durumda ya modeli biraz daha eğitmeliyiz ya da learning rate'i arttırmalıyız.
- Bir diğer durum: too few epochs durumudur. Yine **training loss > valid loss** durumu ile karşılaşırız yani too few epochs ile too low lr durumları birbirine çok benzer sonuçta ikisinde de modeli optimize edememiş oluyoruz.
- Bu durumda daha fazla epoch ile modeli eğitmeyi deneyebiliriz, eğer bu çok uzun sürecek gibi görünüyorsa, higher learning rate deneyebiliriz.
- Soru: Tamam yukarıdaki iki durumdada model optimize olamıyor onu anladım, ancak nasıl oluyor da validation set üzerinde daha iyi performans gösteriyor? Bu şans mı yoksa hep böyle mi olur? Her zaman böyle olması bana anlamsız geliyor, belki şans

eseri, başarısız optimizasyon ile elde edilen model sadece o validation set için iyi bir performans gösterebilir ancak bunun bir sürekliliği olacağını sanmıyorum.

Biri training loss>valid loss durumunun olası sebepleri için şunları yazmış:

- your validation set contains samples very close to some of the train set
- your validation set is simpler (e.g. you oversample hard cases for training)
- using any form of dropout may result in validation loss (when dropout is not applied) less than train loss (when dropout is applied)
- applying heavy augmentations to train data but not validation

- Bir diğer olası problem ise **too many epochs.** Bu durumda, overfitting ile karşılaşabiliriz. Jeremy diyor ki sanılanın aksine deep learning ile overfit etmek çok zor, teddy bear örneği için, tüm data augmentation'I, dropout'ı, weight decay'I vesaire herşeyi kapattık buna rağmen model zar zor overfit etti belki de etmedi diyor.

- Jeremy bu kısımda çok önemli bir şey diyor, şimdiye kadar bildiğin şey yanlış diyor training loss < validation loss demek overfit demek değil diyor. Bununz zaten olması gerek diyor, bu kötü bir şey değil diyor. Overfitting'i gösteren şey, validation error'ın zamanla kötüye gitmesi diyor.

- Yani şunu diyor, benim training loss'um < validation loss olsa da eğer zamanla validation loss'um azalıyorsa bu neden problem olsun ki diyor. Sonuçta zaten yapmaya çalıştığım şey bu, modelimin error rate'ini düşürmek, eğer modelimin error rate'i düşmüyorsa işte o zaman problem var demektir.

- Soru:Ne kadar data gerektiğini nasıl bileceğiz?

- Kesin bir cevabı yok önce deneriz, sonuçtan memnun değilsek artırırız. Eğer learning rate seçimimiz ve batch number'ımız doğru ise (lr'yi artırınca massive losses ile düşürünce ise very slow learning ile karşılaşıyorsak ve batch number artınca error kötüye gidiyorsa), fakat hala istediğimiz sonucu alamıyorsak, daha sonra göreceğimiz bazı seçenekler var bunlardan biri daha fazla data kullanmak, böylece daha uzun süre train edebiliriz ve higher accuracy elde edebiliriz.

- Soru:Unbalanced data söz konusuysa ne yapmalıyız? (mesela class1'den 200 örnek varken class2'den 50 örnek varsa)

- Cevap:Hiçbirşey yapmana gerek yok, usual practice works just fine. Bir paper'da az örnek olan class'ın datasını alıp kopyalamak iyi olabilir diyor, buna oversampling denir. Ancak pratikte Jeremy buna hiç gerek duymamış.

4. Şimdi single variant linear regression problemini pytorch ile inceleyeceğiz.

- Yani temelde y = a0*x0 + a1*x1 denkleminde x0'ı 1 kabul edeceğiz ve x1'e karşılık y'yi veren fonksiyonu yani bir başka deyişle a0 ve a1'i veya bir başka deyişle doğrunun bias'ını ve eğimini bulacağız.
- Bildiğimiz linear regression problemini gösteriyor. Ancak bu problemi pytorch ile çözeceğiz böylece yavaştan pytorch'u anlamaya başlayacağız.