APPLIED RANDOM FOREST

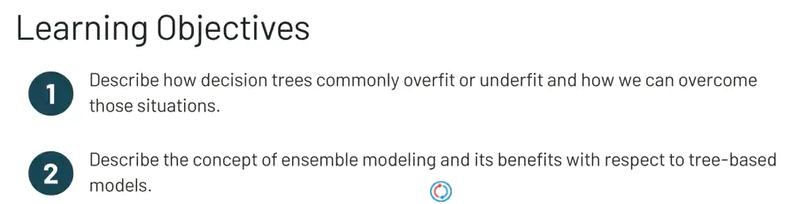
Lesson Introduction

Welcome to Lesson Two, applied random forest. [İkinci Ders'e hoş geldiniz, uygulamalı rasgele orman.]

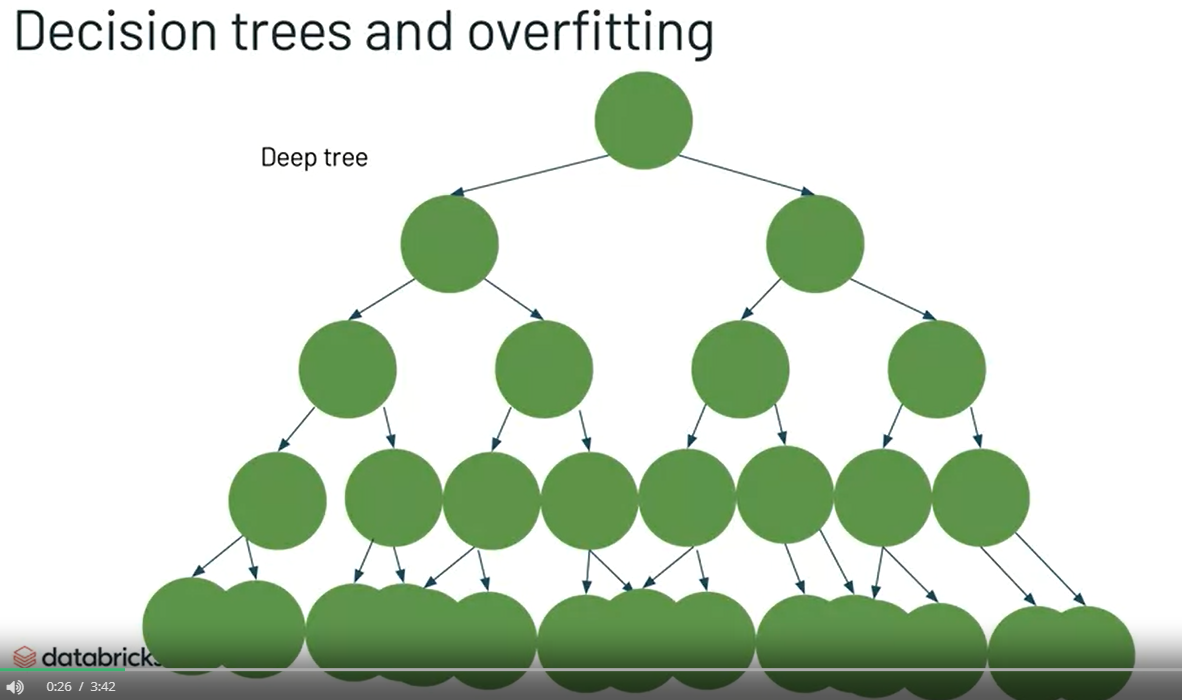
In this lesson, we will use a regression problem to demonstrate how random forest can be advantageous over decision trees with respect to model generalization. [Bu derste, model genellemesi açısından rastgele ormanın karar ağaçlarına göre nasıl avantajlı olabileceğini göstermek için bir regresyon problemi kullanacağız.]

We'll learn that this is due to a technique called ensembling that combines the power of single decision trees into a more robust model. [Bunun, tek karar ağaçlarının gücünü daha sağlam bir modelde birleştiren, birleştirme adı verilen bir teknikten kaynaklandığını öğreneceğiz.]

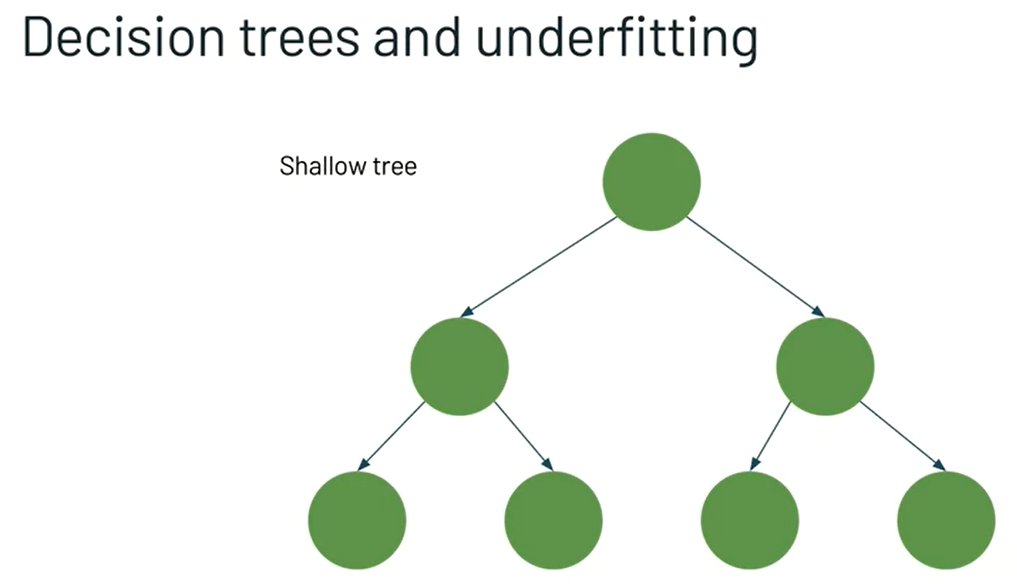
# Introduction to Ensemble Modeling



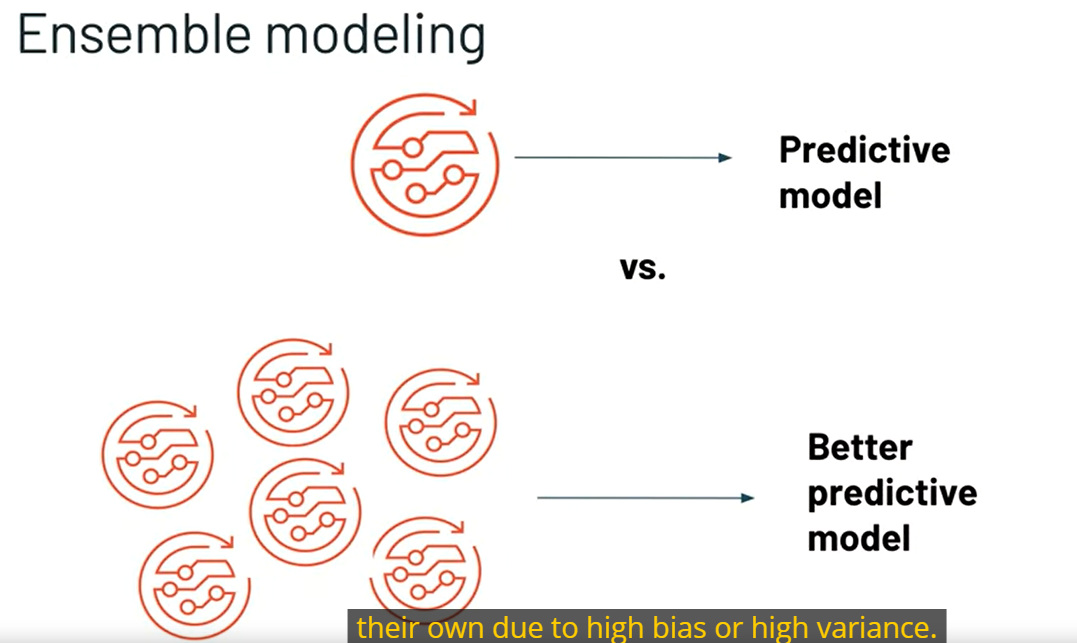
In this video, we'll describe how decision trees usually overfit or underfit to data, and how we can overcome those situations. [Bu videoda, karar ağaçlarının genellikle verilere nasıl fazla ya da eksik uyduğunu ve bu durumların üstesinden nasıl gelebileceğimizi anlatacağız.] We'll then talk about the concept of ensemble modeling and its benefits with respect to tree-based models. [Daha sonra topluluk modelleme kavramı ve ağaç tabanlı modellere göre faydalarından bahsedeceğiz.]



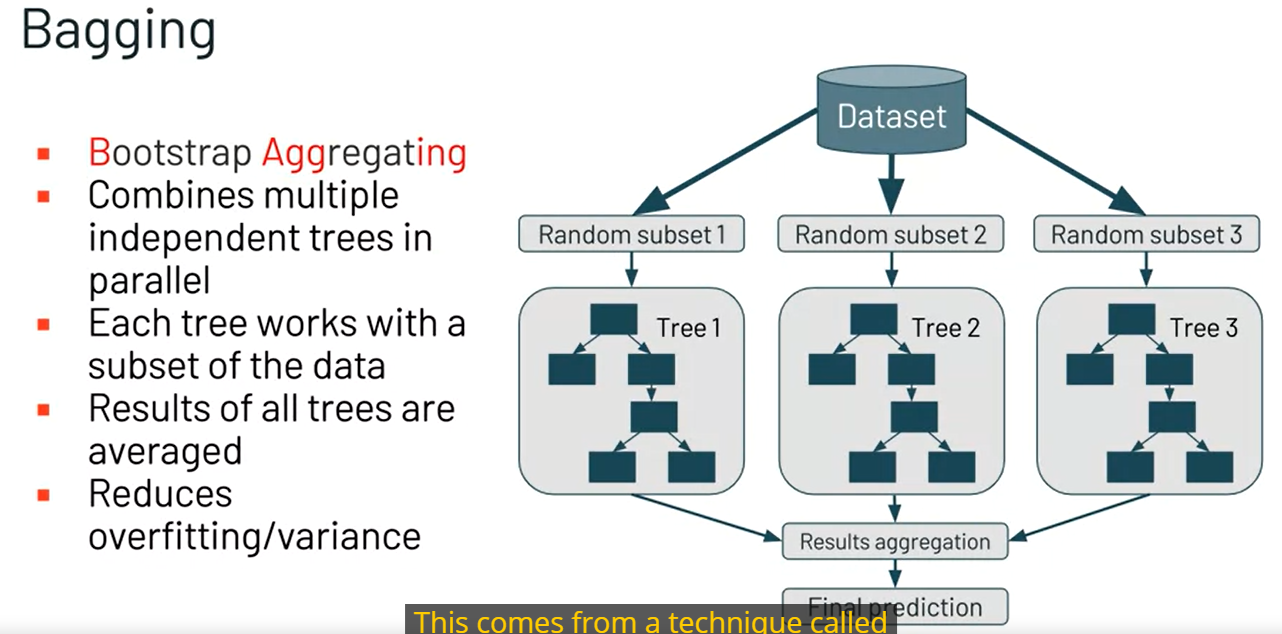
As we learned in the previous lesson, if we don't exercise any prunings strategies, a decision tree will learn the training dataset perfectly and achieve 100 percent accuracy. [Önceki derste öğrendiğimiz gibi, herhangi bir budama stratejisi uygulamazsak, bir karar ağacı eğitim veri setini mükemmel bir şekilde öğrenecek ve yüzde 100 doğruluk elde edecektir.] Even with pre-pruning, a single decision tree tends to overfit to the training data, leading to poor performance on the test set or a new data. [Ön budama ile bile, tek bir karar ağacı eğitim verilerine fazla uyma eğilimi gösterir ve bu da test setinde veya yeni bir veride düşük performansa yol açar.] This means that the model will have low bias and high variance. [Bu, modelin düşük yanlılığa ve yüksek varyansa sahip olacağı anlamına gelir.] It learns the training dataset really well, but it fails to accurately predict new data. [Eğitim veri setini gerçekten iyi öğrenir, ancak yeni verileri doğru bir şekilde tahmin edemez.]



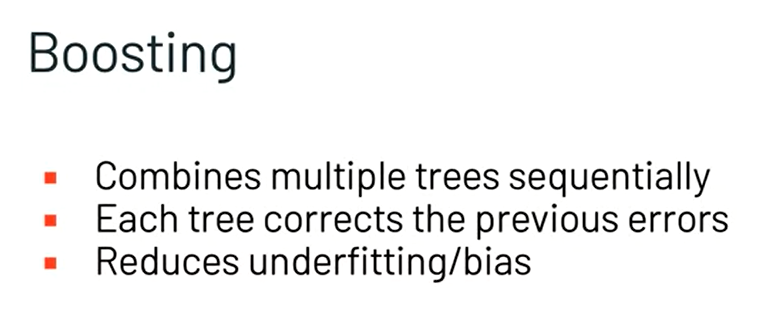
On the other hand, some pruning methods like limiting tree depth can lead to very shallow trees, sometimes as much as only two levels of splitting. [Öte yandan, ağaç derinliğini sınırlamak gibi bazı budama yöntemleri, bazen sadece iki seviyedeki kadar çok sığ ağaçlara yol açabilir.] This can cause the tree to be under-fit, meaning it has high bias and it doesn't learn the training dataset. [Bu, ağacın yetersiz olmasına neden olabilir, yani yüksek önyargıya sahiptir ve eğitim veri setini öğrenmez.]



A very effective method for addressing this problem by reducing bias and variance is called ensemble modeling. [Önyargı ve varyansı azaltarak bu sorunu çözmek için çok etkili bir yönteme topluluk modellemesi denir.] Ensembles are simply combinations of multiple machine learning models that lead to more powerful performance than individual models. [Topluluklar, tek tek modellerden daha güçlü performansa yol açan birden çok makine öğrenimi modelinin basit bir birleşimidir.] The individual models that form the building blocks of these ensemble models are often weak learners that don't perform well on their own due to high bias or high variance. [Bu topluluk modellerinin yapı taşlarını oluşturan bireysel modeller, genellikle yüksek önyargı veya yüksek varyans nedeniyle kendi başlarına iyi performans göstermeyen zayıf öğrenicilerdir.] By combining multiple weak learners, we can create a stronger, more accurate model. [Birden fazla zayıf öğrenciyi birleştirerek daha güçlü ve daha doğru bir model oluşturabiliriz.] There are various types of ensemble methods that work with a variety of datasets and problems. [Çeşitli veri kümeleri ve problemlerle çalışan çeşitli topluluk yöntemleri vardır.] Two that use decision trees as their base models are called bagging and boosting. [Temel modelleri olarak karar ağaçlarını kullanan iki tanesine torbalama ve artırma denir.] Both of these methods result in models with significantly better performance than single decision trees. [Bu yöntemlerin her ikisi de, tekli karar ağaçlarından önemli ölçüde daha iyi performansa sahip modeller ile sonuçlanır.]

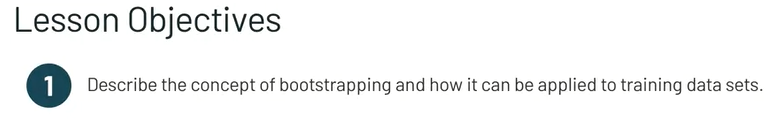


Bagging is a term coined from bootstrap aggregating. [Torbalama, önyükleme toplamadan türetilmiş bir terimdir.] This comes from a technique called bootstrapping and can be applied to decision trees. [Bu, önyükleme adı verilen bir teknikten gelir ve karar ağaçlarına uygulanabilir.] Bagging combines multiple independent trees in parallel into one model. [Torbalama, birden fazla bağımsız ağacı paralel olarak tek bir modelde birleştirir.] Instead of training each tree on all of the features in the training dataset, each tree is trained on a sub-sample of the training dataset. [Her ağacı eğitim veri kümesindeki tüm özellikler üzerinde eğitmek yerine, her ağaç eğitim veri kümesinin bir alt örneğinde eğitilir.] This results in each tree being different from the others, and each tree might overfit the training data in different ways. [Bu, her ağacın diğerlerinden farklı olmasına neden olur ve her ağaç, eğitim verilerini farklı şekillerde aşabilir.] By averaging their results into one ensemble model, we can reduce overfitting and thus variance, while retaining predictive accuracy. [Sonuçlarının ortalamasını tek bir grup modelinde toplayarak, tahmin doğruluğunu korurken fazla uyumu ve dolayısıyla varyansı azaltabiliriz.]

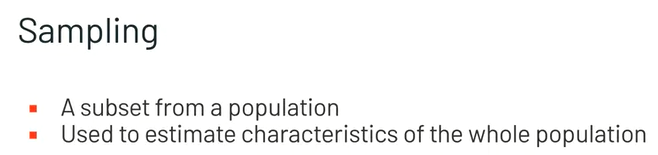


Boosting is a similar concept to bagging, but instead of combining multiple independent decision trees in parallel, it combines trees sequentially into one model. [Hızlandırma, torbalamaya benzer bir kavramdır, ancak birden fazla bağımsız karar ağacını paralel olarak birleştirmek yerine, ağaçları sırayla tek bir modelde birleştirir.] The trees in a boosting ensemble model are not independent of each other. [Destekleyici topluluk modelindeki ağaçlar birbirinden bağımsız değildir.] Each tree attempts to correct the errors of the previous tree. [Her ağaç bir önceki ağacın hatalarını düzeltmeye çalışır.] The resulting ensemble model will have less underfitting and bias than the individual weak learner models or decision trees. [Ortaya çıkan topluluk modeli, bireysel zayıf öğrenci modellerinden veya karar ağaçlarından daha az uyumsuzluğa ve önyargıya sahip olacaktır.] Determining which of these two methods to use, bagging or boosting, depends on understanding the bias-variance trade-off and how it applies to your data and objectives. [Torbalama veya artırma olmak üzere bu iki yöntemden hangisinin kullanılacağını belirlemek, önyargı-varyans dengesini ve bunun verileriniz ve hedefleriniz için nasıl uygulandığını anlamaya bağlıdır.] You will now complete a short quiz to check your understanding of these ensembling methods. [Şimdi bu birleştirme yöntemlerini ne kadar anladığınızı kontrol etmek için kısa bir testi tamamlayacaksınız.] Then in the next video, we'll talk about the concept of bootstrapping data. [Ardından bir sonraki videoda, önyükleme verileri kavramından bahsedeceğiz.]

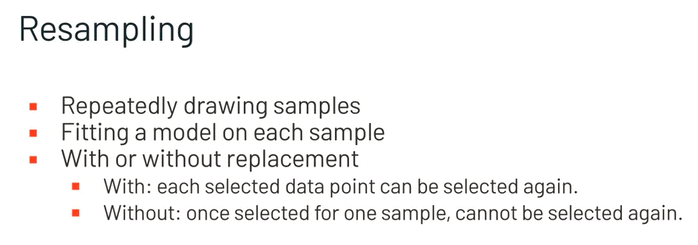
Bootstrap Sampling Training Data



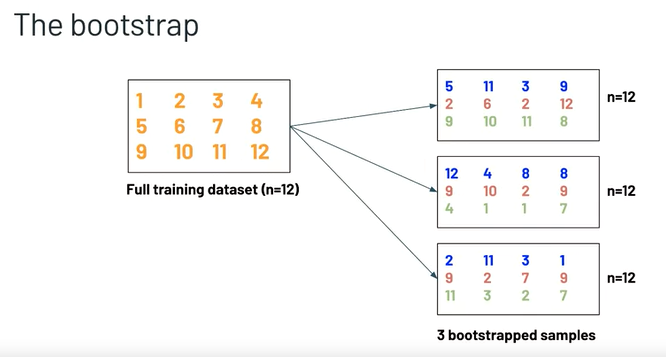
In this video, we'll describe the concept of bootstrapping, including the idea of sampling, sampling with replacement, and explore the resulting variation in datasets. [Bu videoda, örnekleme, değiştirme ile örnekleme fikri dahil olmak üzere önyükleme kavramını açıklayacağız ve sonuçta ortaya çıkan veri kümelerindeki varyasyonu keşfedeceğiz.]



You may remember that sampling refers to the process of selecting a subset of individuals from within a statistical population in order to estimate characteristics of the whole population. [Örneklemenin, tüm popülasyonun özelliklerini tahmin etmek için istatistiksel bir popülasyon içinden bireylerin bir alt kümesini seçme sürecini ifade ettiğini hatırlayabilirsiniz.] When we apply the concept of sampling to machine learning models, it means taking a subset of our data, which is the training set, and building a model to learn characteristics of the entire dataset from this sample. [Örnekleme kavramını makine öğrenme modellerine uyguladığımızda, bu, eğitim seti olan verilerimizin bir alt kümesini almak ve bu örnekten tüm veri kümesinin özelliklerini öğrenmek için bir model oluşturmak anlamına gelir.]



Resampling means repeatedly drawing samples from a training dataset and fitting the model on each sample. [Yeniden örnekleme, bir eğitim veri kümesinden tekrar tekrar örnek alınması ve modelin her bir örneğe uydurulması anlamına gelir.] It can be done with or without replacement. [Değiştirme ile veya değiştirmeden yapılabilir.] With replacement just means that each selected data point can be selected again. [Değiştirme ile sadece seçilen her veri noktasının tekrar seçilebileceği anlamına gelir.] It's as if you're pulling a sample out of a bag and then putting it back in. [Sanki bir torbadan bir numune alıp sonra tekrar koyuyormuşsunuz gibi.] Therefore, each selection is independent of the previous selections. [Bu nedenle, her seçim önceki seçimlerden bağımsızdır.] Sampling without replacement means that once the sample is selected, it can't be selected again. [Değiştirmeden örnekleme, örneğin bir kez seçildikten sonra tekrar seçilemeyeceği anlamına gelir.] Each resample is affected by the previous ones, so they are not independent. [Her yeniden örnekleme öncekilerden etkilenir, bu nedenle bağımsız değildirler.]

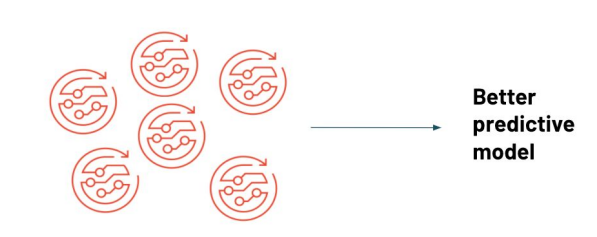


One of the most common methods of resampling is called the bootstrap. [Yeniden örneklemenin en yaygın yöntemlerinden birine önyükleme denir.] The bootstrap does random sampling with replacement. [Önyükleme, değiştirme ile rastgele örnekleme yapar.] Samples are generally of the same size of the full training dataset. [Örnekler genellikle tam eğitim veri kümesiyle aynı boyuttadır.] As we see here, where we have three bootstrap samples that are 512, n equals 12, just like the full training dataset. [Burada gördüğümüz gibi, tam eğitim veri kümesi gibi, 512, n eşittir 12 olan üç önyükleme örneğimiz var.] A bootstrap sample can have duplicate observations when we sample with replacement because we're essentially drawing a record or a number out of our full training dataset, recording it into our bootstrap sample, and then putting it back into the pool so it can be sampled again. [Bir önyükleme örneği, değiştirme ile örneklediğimizde yinelenen gözlemlere sahip olabilir, çünkü esas olarak tam eğitim veri kümemizden bir kayıt veya bir sayı çiziyoruz, onu önyükleme örneğimize kaydediyoruz ve sonra örneklenebilmesi için havuza geri koyuyoruz. Yeniden.] We see that here on the slide, where we have repetitions of certain numbers in each bootstrap sample. [Bunu, her bir önyükleme örneğinde belirli sayıların tekrarlarının olduğu slaytta görüyoruz.] For example, the sample on the top has two nines and two elevens. [Örneğin, üstteki örnekte iki dokuzluk ve iki onbir var.] We can have the same number repeated more than once, as you can see in the example of three bootstrapped samples on the right from the full training dataset. [Tam eğitim veri kümesinden sağdaki üç önyüklenmiş örnek örneğinde görebileceğiniz gibi, aynı sayıyı bir kereden fazla tekrarlayabiliriz.] This technique increases the variability in a training dataset while not altering the fundamental characteristics of the dataset. [Bu teknik, bir eğitim veri kümesindeki değişkenliği artırırken, veri kümesinin temel özelliklerini değiştirmez.] This means that a model trained on bootstrap samples will generalize better to new data. [Bu, önyükleme örnekleri üzerinde eğitilmiş bir modelin yeni verilere daha iyi genelleştireceği anlamına gelir.] In other words, this method decreases the variance of a model. [Başka bir deyişle, bu yöntem bir modelin varyansını azaltır.]

# Aggregating Bootstrapped Results

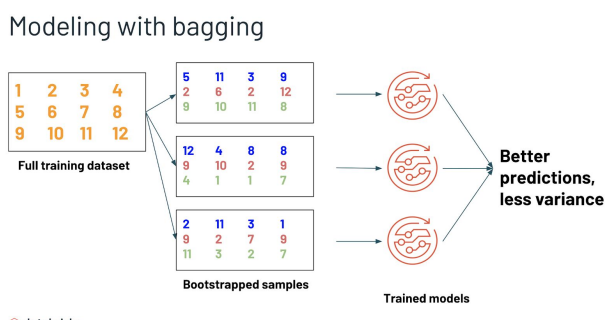
We learned about the bootstrap method in the last video. In general, bootstrap aggregating (bagging) is a method for reducing the variance of a model: it is especially useful when applied to decision trees that often suffer from high variance - involves taking many sampled/bootstrapped training data sets, building separate models on each, and averaging the resulting predictions, to obtain a model with lower variance. This works because there is variability in the training dataset, and therefore the fitted model is also subject to variability: if we had trained the model on another subsampled dataset from the population, we would have obtained a different model with different results

Son videoda bootstrap yöntemini öğrenmiştik. Genel olarak, bootstrap aggregating (bagging) , bir modelin varyansını azaltmak için bir yöntemdir: özellikle genellikle yüksek varyanstan muzdarip decision trees e uygulandığında yararlıdır - birçok sampled/bootstrapped eğitim veri seti almayı, her biri üzerinde ayrı modeller oluşturmayı içerir. ve daha düşük varyanslı bir model elde etmek için elde edilen tahminlerin ortalamasını almak. Bu işe yarar çünkü eğitim veri kümesinde değişkenlik vardır ve bu nedenle fitted model de değişkenliğe tabidir: modeli popülasyondan başka bir alt örneklenmiş veri kümesi üzerinde eğitmiş olsaydık, farklı sonuçlara sahip farklı bir model elde ederdik.



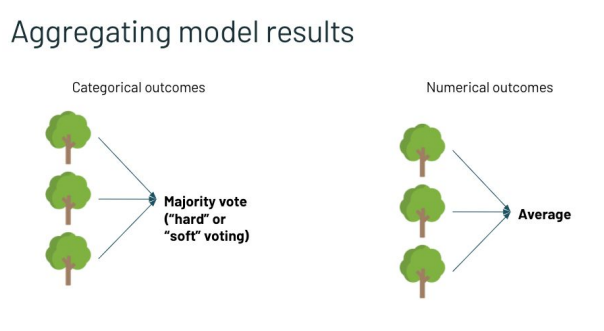
But we don’t usually have multiple training data sets - so we can artificially create them with bootstrapping. And then, we aggregate the results from the bootstrapped samples to arrive at better predictions.

Ancak genellikle birden fazla eğitim veri setimiz yoktur - bu nedenle bunları bootstrapping ile yapay olarak oluşturabiliriz. Ardından, daha iyi tahminlere ulaşmak için bootstrapping örneklerden elde edilen sonuçları topluyoruz.



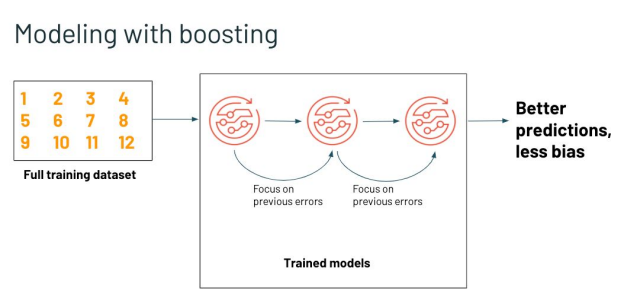
How do we actually aggregate the results of different models into one output or result? For categorical there are two methods - with “hard voting”, we take the majority vote of all the models and assign the unknown data point to that class. Another method, called soft voting, is where we consider the probabilities of each class returned by all the models, average these probabilities and keep the class with the highest average probability. For numerical, average the results of each individual model and apply that to the unknown data point to get the result of the ensemble model.

Farklı modellerin sonuçlarını gerçekte nasıl tek bir çıktı veya sonuç olarak topluyoruz? Kategorik için iki yöntem vardır - “hard voting” ile tüm modellerin çoğunluk oyu alır ve bilinmeyen veri noktasını o sınıfa atarız. soft voting adı verilen diğer bir yöntem, tüm modeller tarafından döndürülen her bir sınıfın olasılıklarını dikkate aldığımız, bu olasılıkların ortalamasını aldığımız ve sınıfı en yüksek ortalama olasılığa sahip tuttuğumuz yöntemdir. Sayısal olarak, her bir modelin sonuçlarının ortalamasını alın ve ensemble modelinin sonucunu elde etmek için bunu bilinmeyen veri noktasına uygulayın.



We mentioned boosting as another type of ensemble model in a previous video. Different from bagging - instead of combining multiple independent trees in parallel, it iteratively combines trees sequentially into one model. The trees are not independent - each tree attempts to correct the errors of the previous tree, by giving more weight to observations in the dataset that were incorrectly predicted by the previous models in the sequence. The resulting ensemble model will have less underfitting and bias than the individual weak learner models. One commonly used boosting technique is Gradient Boosting Trees. Another is called xgboost. These are often used to win data science competitions.

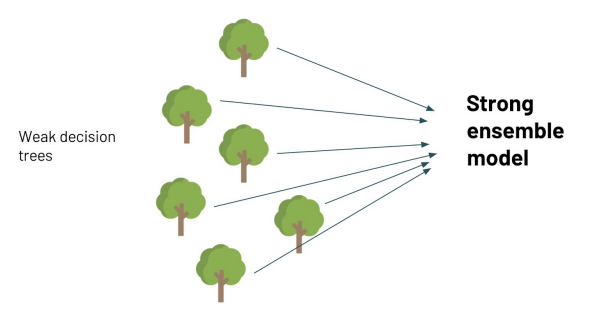
Bir önceki videoda bir diğer ensemble modeli olarak boosting bahsetmiştik. bagging farklı olarak - birden fazla bağımsız ağacı paralel olarak birleştirmek yerine, ağaçları ardışık olarak tek bir modelde yinelemeli olarak birleştirir. Ağaçlar bağımsız değildir - her ağaç, dizideki önceki modeller tarafından yanlış tahmin edilen veri kümesindeki gözlemlere daha fazla ağırlık vererek önceki ağacın hatalarını düzeltmeye çalışır. Ortaya çıkan ensemble modeli, bireysel zayıf öğrenci modellerinden daha az underfitting ve bias sahip olacaktır. Yaygın olarak kullanılan bir boosting tekniği Gradient Boosting Trees'dir. Bir başkasına xgboost denir. Bunlar genellikle veri bilimi yarışmalarını kazanmak için kullanılır.



# Random Forest Algorithm

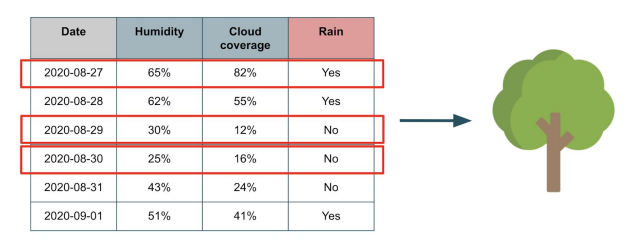
Random forest is an ensemble modeling algorithm that combines multiple weak decision trees into a strong learner. The individual trees in any ensemble algorithm can be either shallow (not many depths of decisions) or deep (lots of depths). Shallow trees have less variance but higher bias, so they are generally a better choice for boosting or sequential methods that decrease bias. Conversely, Deep trees have low bias but high variance and are good for bagging or parallel methods that result in lower variance.

Random forest, birden çok zayıf decision tree güçlü bir öğrenicide birleştiren bir ensemble modelleme algoritmasıdır. Herhangi bir ensemble algoritmasındaki tek tek ağaçlar ya sığ (çok fazla karar derinliği yok) ya da derin (birçok derinlik) olabilir. Sığ ağaçların varyansı daha az, ancak bias daha yüksektir, bu nedenle genellikle bias azaltan boosting veya sıralı yöntemler için daha iyi bir seçimdir. Tersine, Derin ağaçlar düşük bias sahiptir ancak varyansı yüksektir ve daha düşük varyansla sonuçlanan bagging veya paralel yöntemler için iyidir.



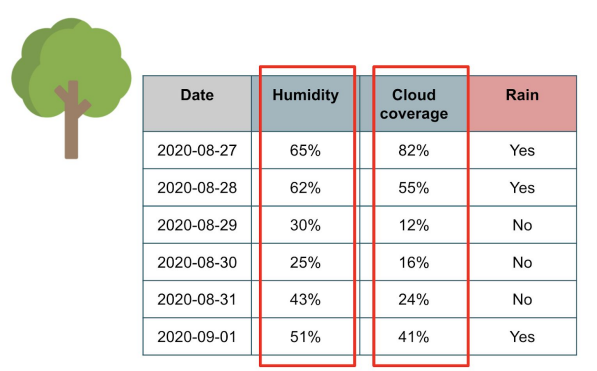
Random forest combines deep trees, fitted on bootstrap samples, to reduce variance. In addition to using bootstrapped samples for each individual decision tree, random forest employs a technique that makes each tree less correlated with the others and introduces randomness. When fitting each tree, a sub sample is taken from the observations, or rows, in the dataset, with replacement so that each subsample can have repetitions.

Random forest, varyansı azaltmak için boosting örneklerine fit edilen derin ağaçları birleştirir. Random forest, her bir karar ağacı için bootstrapped örneklerin kullanılmasına ek olarak, her bir ağacı diğerleriyle daha az ilişkilendiren ve rastgelelik getiren bir teknik kullanır. Her bir ağacı yerleştirirken, veri kümesindeki gözlemlerden veya satırlardan bir alt örnek alınır ve her bir alt örneğin tekrarları olabilir.



In addition to the randomness introduced by sampling from the observations in the training dataset, within each decision tree, at every split or node the tree only considers a randomly selected subset of the features.

Eğitim veri kümesindeki gözlemlerden örnekleme yoluyla getirilen rastgeleliğe ek olarak, her decision tree içinde, her split veya node ağaç, özelliklerin yalnızca rastgele seçilmiş bir alt kümesini dikkate alır.



These two techniques are where the randomness in random forest comes from, and ensures that each of the decision trees in the forest are different, thus reducing variance. There are some hyperparameters that are specific to random forest, such as the number of trees (or estimators) to build, and the maximum number of features to select from. In addition, the individual tree hyperparameters can be set when instantiating a random forest

Bu iki teknik, random forest rastgeleliğin nereden geldiğidir ve ormandaki decision trees her birinin farklı olmasını sağlayarak varyansı azaltır. Oluşturulacak ağaç (veya tahmin edici) sayısı ve seçilebilecek maksimum özellik sayısı gibi random forest özgü bazı hiperparametreler vardır. Ayrıca, random forest örneği başlatılırken bireysel ağaç hiperparametreleri ayarlanabilir.

# Applied Random Forest

In this demo, we're going to use scikit-learn to create a random forest model. Our first step, again, is going to be preparing our dataset. In this example, we're, again, going to try to predict a customer steps based on their other recorded metrics. We're using the same lifestyle table that we created previously as a pandas DataFrame. Then we're going to set our x and our y with the values that we want. Then we'll do the train test split. The first thing we'll do is to fit a random forest regressor model onto that training set. We'll evaluate the results of that regression model. We see already that with the random forest, instead of the single decision tree, these results are looking fairly significantly better than when we use just a single decision tree. We get a really good score on the training set of 98.7. While still not getting 100 percent because we don't want it to be too overfitted. Then our test set is getting a little bit better. We're almost at 90 percent on the test set.

We can try to improve this by tuning some hyperparameters. Remember that some of the hyperparameters that are available for decision trees can also be tuned in random forests. These will just be applied to each of the individual decision trees, which are also called estimators that comprise the random forest. In addition to that, there's also some hyperparameters that are specific to the random forest itself. For example, we have this bootstrap option, which is a Boolean value, and the default is set to true. Now, this determines whether bootstrapped samples are used when you're building each individual tree. If you set that to false, it means that every single tree uses the entire dataset. You would probably never want to change that, we're just pointing out that, that is an option that can be changed. But if you think a little bit about it, if you set that to false and you weren't doing a bootstrap sample for each individual tree, you would lose that randomness where each tree is looking at a different sample.

Now, we're going to try tuning some of our hyperparameters. N\_estimators is the number of estimators or the number of trees in your forest. We'll try setting this to 50. I believe the default is 100, and see if it does worse or better if we use fewer trees. Sometimes you might adjust this based on how long it's going to take to run the model. We'll set a max depth of eight and that applies to each individual decision tree. Then you don't have to specify the bootstrap here because the true is already at the default. But just so that it's really clear that's what we're doing, we'll include that here. Now, we'll see how we did. We see it's getting slightly better. The previous score is up here. We're a little bit more spread apart between the training and test sets. Now, we're getting a little bit lower training set, but a slightly higher test set. We see that we've controlled a little bit of the overfitting and also decreased the variance just slightly. Now, let's look at what happens if we set bootstrap to false and use the entire dataset for each tree.

Now, we see that our performance got worse for both the training and the test set. As expected, because like I said, if we're not using bootstrap samples, we eliminate some of that randomness. So the advantages of random forests are somewhat taken away by not having that randomness within the trees.

# Applied Random Forest Lab

Next, you'll complete a lab on applied random forest using Databricks.

To do so:

1. Log in to your Databricks Community Edition account.

2. Locate your ADSDA folder.

3. Open Module 4, Lesson 2.

4. Click on 4.2.2 Lab - Applied Random Forest to open your notebook.

Note: If you haven’t already imported the course DBC file, review instructions in Module 1's "Hands-on with Databricks" video.

Remember that if you need help with this lab, the solution to the lab has been included in the course materials file.