# ML.Net Samples Notes

Jan ‘19

D:\Projects\back\MLNet

Machine Learning

<https://blogs.msdn.microsoft.com/dotnet/2019/01/10/announcing-ml-net-0-9-machine-learning-for-net/>

Written 10Jan’19

ML.NET is an open-source and cross-platform machine learning framework (Windows, Linux, macOS) for .NET developers. Using ML.NET, developers can leverage their existing tools and skillsets to develop and infuse custom AI into their applications by creating custom machine learning models.

ML.NET allows you to create and use machine learning models targeting common tasks such as classification, regression, clustering, ranking, recommendations and anomaly detection. It also supports the broader open source ecosystem by proving integration with popular deep-learning frameworks like TensorFlow and interoperability through ONNX. Some common use cases of ML.NET are scenarios like Sentiment Analysis, Recommendations, Image Classification, Sales Forecast, etc. Please see our samples for more scenarios.

# ML.Net Version 0.9

<https://blogs.msdn.microsoft.com/dotnet/2019/01/10/announcing-ml-net-0-9-machine-learning-for-net/>

New features……..

## Community Samples

<https://github.com/dotnet/machinelearning-samples/blob/master/docs/COMMUNITY-SAMPLES.md>

# Feature Contribution Calculation (FCC)

New in V0.9

<https://github.com/dotnet/machinelearning/blob/master/docs/samples/Microsoft.ML.Samples/Dynamic/FeatureContributionCalculationTransform.cs>

The Feature Contribution Calculation (FCC for short) shows which features are most influential for a model’s prediction on a particular and individual data sample by determining the amount each feature contributed to the model’s score for that particular data sample.

FCC is particulary important when you initialy have a lot of features/attributes in your historic data and you want to select and use only the most important features because using too many features (especially if including features that don’t influence the model) can reduce the model’s performance and accuracy. Therefore, with FCC you can identify the most influential positive and negative contributions from the initial attribute set.

You can use FCC to produce feature contributions with code like the following: blah blah blah

# Getting the code

<https://github.com/dotnet/machinelearning> to D:\Projects\back\MLNet\machinelearning

# Getting the samples

<https://dotnet.microsoft.com/apps/machinelearning-ai/ml-dotnet>

<https://github.com/dotnet/machinelearning-samples> to D:\Projects\back\MLNet\MlSamples

# ML.Net Tutorial Guide

<https://dotnet.microsoft.com/learn/machinelearning-ai/ml-dotnet-get-started-tutorial>

<https://docs.microsoft.com/en-us/dotnet/machine-learning/tutorials/>

The complete set of samples from <https://github.com/dotnet/machinelearning-samples> are here: D:\Projects\back\MLNet\dotnetSamples

The ones I’m playing with are here D:\Projects\back\MLNet\jplourdMLSamples

The following tutorials enable you to understand how to use [ML.NET](https://docs.microsoft.com/en-us/dotnet/machine-learning/index) to build custom machine learning solutions and integrate them into your .NET applications:

* [Sentiment analysis](https://docs.microsoft.com/en-us/dotnet/machine-learning/tutorials/sentiment-analysis): demonstrates how to apply a **binary classification** task using ML.NET.
* [Taxi fare predictor](https://docs.microsoft.com/en-us/dotnet/machine-learning/tutorials/taxi-fare): demonstrates how to apply a **regression** task using ML.NET.
* [Iris clustering](https://docs.microsoft.com/en-us/dotnet/machine-learning/tutorials/iris-clustering): demonstrates how to apply a **clustering** task using ML.NET.

For more examples that use ML.NET, check the [dotnet/machinelearning-samples](https://github.com/dotnet/machinelearning-samples) GitHub repository.

## Sentiment Analysis Binary Classification

<https://docs.microsoft.com/en-us/dotnet/machine-learning/tutorials/sentiment-analysis>

D:\Projects\back\MLNet\MlSamples\machinelearning-samples\samples\csharp\getting-started\BinaryClassification\_SentimentAnalysis\SentimentAnalysis-Solution.sln

I’m confused as to how the projects are organized, there seem to be samples and then a separate tutorial series……….

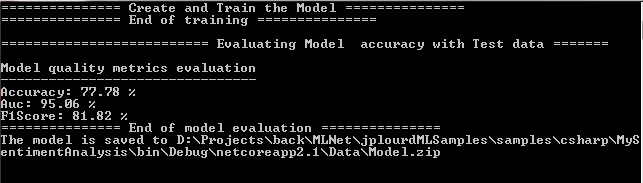
<https://github.com/dotnet/samples> contains a **machine-learning/tutorials** directory

I’ve cloned it here D:\Projects\back\MLNet\dotnetSamples

The code for this sentiment project is here: D:\Projects\back\MLNet\dotnetSamples\samples\machine-learning\**tutorials**\SentimentAnalysis

This version works and the one I’m building by following along with the tutorial does as well (D:\Projects\back\MLNet\jplourdMLSamples\samples\csharp\**MySentimentAnalysis**\Program.cs)

The output:



***Classification*** is a machine learning task that uses data to **determine** the category, type, or class of an item or row of data. For example, you can use classification to:

* Identify sentiment as positive or negative.
* Classify email as spam, junk, or good.
* Determine whether a patient's lab sample is cancerous.
* Categorize customers by their propensity to respond to a sales campaign.

Classification tasks are frequently one of the following types:

* Binary: either A or B.
* Multiclass: multiple categories that can be predicted by using a single model.

SentimentData is the input dataset class and has a float (Sentiment) that has a value for sentiment of either positive or negative, and a string for the comment   
(SentimentText). Both fields have Column attributes attached to them. This attribute describes the order of each field in the data file, and which is the Label field.   
SentimentPrediction is the class used for prediction after the model has been trained. It has a single boolean (Sentiment) and a PredictedLabel ColumnName attribute. The   
Label is used to create and train the model, and it's also used with a second dataset to evaluate the model. The PredictedLabel is used during prediction and evaluation. For   
evaluation, an input with training data, the predicted values, and the model are used.

ML.NET's transform pipelines compose a custom set of transforms that are applied to your data before training or testing. The transforms' primary purpose is   
data [featurization](https://docs.microsoft.com/en-us/dotnet/machine-learning/resources/glossary" \l "feature-engineering). Machine learning algorithms understand [featurized](https://docs.microsoft.com/en-us/dotnet/machine-learning/resources/glossary" \l "feature) data, so the next step is to transform our textual data into a format that our ML   
algorithms recognize. That format is a [numeric vector](https://docs.microsoft.com/en-us/dotnet/machine-learning/resources/glossary#numerical-feature-vector).

Next, call mlContext.Transforms.Text.FeaturizeText which featurizes the text column (SentimentText) column into a numeric vector   
called Features used by the machine learning algorithm. This is a wrapper call that returns an [EstimatorChain<TLastTransformer>](https://docs.microsoft.com/en-us/dotnet/api/microsoft.ml.runtime.data.estimatorchain-1) that will   
effectively be a pipeline. Name this pipeline as you will then append the trainer to the EstimatorChain.

To add the trainer, call the mlContext.Transforms.Text.FeaturizeText wrapper method which returns a [FastTreeBinaryClassificationTrainer](https://docs.microsoft.com/en-us/dotnet/api/microsoft.ml.trainers.fasttree.fasttreebinaryclassificationtrainer) object. This is   
a decision tree learner you'll use in this pipeline. The FastTreeBinaryClassificationTrainer is appended to the pipeline and accepts the   
featurized SentimentText (Features) and the Label input parameters to learn from the historic data.

You train the model, [TransformerChain<TLastTransformer>](https://docs.microsoft.com/en-us/dotnet/api/microsoft.ml.data.transformerchain-1), based on the dataset that has been loaded and transformed. Once the estimator has been   
defined, you train your model using the [Fit](https://docs.microsoft.com/en-us/dotnet/api/microsoft.ml.runtime.data.estimatorchain-1.fit)while providing the already loaded training data. This returns a model to use for   
predictions. pipeline.Fit() trains the pipeline and returns a Transformer based on the DataView passed in. The experiment is not executed until this   
happens.

## TaxiFare regression predictor

D:\Projects\back\MLNet\jplourdMLSamples\samples\csharp\**MyTaxiFarePrediction**

D:\Projects\back\MLNet\dotnetSamples\samples\machine-learning\tutorials\TaxiFarePrediction

Note: ‘Column’ names by convention carry extraordinary meaning!

//When the model is trained and evaluated, by default, the values in the **Label** column are considered as correct values to be predicted.

//As we want to predict the taxi trip fare, copy the FareAmount column into the Label column. To do that,

//use the CopyColumnsEstimator transformation class

var pipeline = mlContext.Transforms.CopyColumns("FareAmount", "**Label**")

//The algorithm that trains the model requires numeric features, so you have **to transform the categorical data**

**//(VendorId, RateCode, and PaymentType) values into numbers**. To do that, use the OneHotEncodingEstimator transformation

// class, which assigns different numeric key values to the different values in each of the columns,

.Append(mlContext.Transforms.Categorical.OneHotEncoding("VendorId"))

.Append(mlContext.Transforms.Categorical.OneHotEncoding("RateCode"))

.Append(mlContext.Transforms.Categorical.OneHotEncoding("PaymentType"))

// The last step in data preparation combines all of the feature columns into the Features column using the

//ColumnConcatenatingEstimator transformation class. By default, **a learning algorithm processes**

**// only features from the Features column**

.Append(mlContext.Transforms.Concatenate("**Features**", "VendorId", "RateCode", "PassengerCount", "TripTime", "TripDistance", "PaymentType"))

## Iris Flowers clustering learner

<https://docs.microsoft.com/en-us/dotnet/machine-learning/tutorials/iris-clustering>

As you don't know to which group each flower belongs to, you choose the unsupervised machine learning task. To divide a data set in groups in such a way that elements in the same group are more similar to each other than to those in other groups, use a clustering machine learning task.

Train on a set of multivalue floats associated with an iris named type (catgory). Given a set of multivalue floats, predict the iris named type.

Data:

5.1,3.5,1.4,0.2,Iris-setosa

4.9,3.0,1.4,0.2,Iris-setosa

…

5.7,2.8,4.5,1.3,Iris-versicolor

6.3,3.3,4.7,1.6,Iris-versicolor

4.9,2.4,3.3,1.0,Iris-versicolor

…

5.8,2.7,5.1,1.9,Iris-virginica

6.8,3.2,5.9,2.3,Iris-virginica

6.7,3.3,5.7,2.5,Iris-virginica

# ML.Net How to guides

<https://docs.microsoft.com/en-us/dotnet/machine-learning/how-to-guides/>

In the How to section of the ML.NET Guide, you can find quick answers to common questions. In some cases, articles may be listed in multiple sections to make them easy to find.

Sections:

Load data

Train models

Evaluate model

Feature engineering

Running the model

# ML Encog: Neural net framework

<https://www.heatonresearch.com/encog/>

Pluralsight course:

<https://app.pluralsight.com/player?course=introduction-to-machine-learning-encog&author=abhishek-kumar&name=introduction-to-machine-learning-encog-m5-intro-encog3&clip=5&mode=live>

D:\Projects\back\MLNet\Encog

Built the core solution to get the runtime dlls

## GitHub

<https://github.com/search?q=encog>

yields 116 repos

Lots of market related examples…….

## Stock Market

<http://www.devx.com/opensource/Article/44014>

PFalkowski

https://github.com/PFalkowski/EncogExperiments

D:\Projects\back\MLNet\Encog\StockMarketSamples

StockDAta proj is non-encog and fetches data. The tests run but what does it do?

EncogExperiments does I know not but attempts to train something…… <https://github.com/PFalkowski/EncogExperiments>

<https://github.com/PFalkowski/MachineLearningResources> Table of ML resources from frameworks to services

### Zotvent prediction using neural net

https://github.com/zotvent/Stock-Prediction-Using-Artificial-Neural-Network

D:\Projects\back\MLNet\Encog\StockMarketSamples\Zotvent.Stock-Prediction

Web UI depictiong actual vs predicted.



How Neural Network works

It takes 3 consecutive days in CSV file. In each day volume and close price were choosen. Then it predicts close price for the 4th day.

Usage

Install NuGet packages mentioned above.

Create database from NNModel.edmx.sql script. If you apply script to existing database make sure that your database is called NeuronNetworkDB.

In API project in Web.config in connectionStrings sector change path to your database.

Log in as administrator and create a new neural network for some company.

Choose created company at home page.

Now you can see the results of trained neural network on chart.

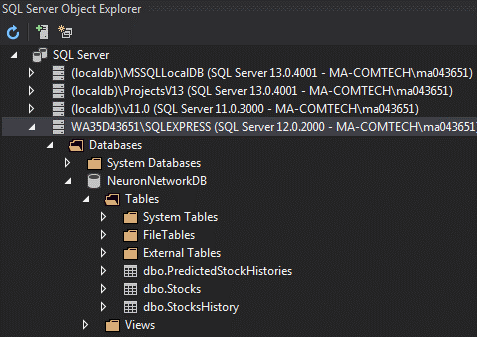
Below the chart you can enter volume and price for 3 consecutive days and make a prediction for the 4th day.

How to log in as administrator

**login: admin**

**password: Qwerty123**

Set up the Microsoft SQL server on my machine. Had to use server explorer panel in Visual Studio to make the connection. Then had the NNModel.edmx thing in SqlObjecWrapper project generate a connection string in the App.config file in that project which I then copied to web.config in the WebAPI project. Only then was I able to run the program and connect to the DB.



<connectionStrings>

<add name="NeuronNetworkDBEntities" connectionString="metadata=res://\*/EntityModel.NNModel.csdl|res://\*/EntityModel.NNModel.ssdl|res://\*/EntityModel.NNModel.msl;provider=System.Data.SqlClient;provider connection string=&quot;data source=WA35D43651\SQLEXPRESS;initial catalog=NeuronNetworkDB;integrated security=True;user id=MA-COMTECH\ma043651;password=blah6k;connect timeout=30;MultipleActiveResultSets=True;App=EntityFramework&quot;" providerName="System.Data.EntityClient" />

</connectionStrings>

# Infer.Net

<https://dotnet.github.io/infer/>

**Infer**.**NET**. **Infer**.**NET** is a framework for running Bayesian inference in graphical models. It can also be used for probabilistic programming as shown in this video. ... **Infer**.**NET**has been used in a wide variety of domains including information retrieval, bioinformatics, epidemiology, vision, and many others.

Translates a model to C# to run on large data set.

In NuGet…………

# Microsoft SQL

Joe lab is sysadm to NeuronNetworkDB

The left side of the image above is the 5-year graph, while the right side is the 1-year graph. In the 5-year graph, the Dow formed a [double bottom](https://stockcharts.com/school/doku.php?id=chart_school:chart_analysis:chart_patterns:double_bottom_reversal) with a target of 21,200 by July 2017, which it did hit. The Dow then formed a [bearish ascending triangle](https://www.investopedia.com/university/charts/charts5.asp) with a target of **21,400 by June 2019**. The [W%R](https://www.investopedia.com/terms/w/williamsr.asp) is -2.08 (extremely overbought), and the [RSI](https://www.investopedia.com/terms/r/rsi.asp) is 61.45 (getting close to overbought). However, in five of the six times the RSI and W%R were at similar levels from 2014-2017, the index rose. Two of the times were in late 2015 and early 2016, which is when I noted that the fall seemed "[largely due to worry about a Chinese slowdown](https://seekingalpha.com/article/3463546-will-market-bottom)." The recent pullback also [correlated](https://www.bloomberg.com/opinion/articles/2018-12-31/china-s-economic-slowdown-is-worsening-stop-dithering-beijing) with worry about a Chinese slowdown. In the 1-year graph, the Dow formed a [bearish triple top](https://stockcharts.com/school/doku.php?id=chart_school:chart_analysis:chart_patterns:triple_top_reversal), with a target of **22,200 by February 2019**. Note that the index fell slightly below the 200-dma three times in the past year and bounced back. The Dow also always bounced back above the 200-dma when falling slightly below it in the 5-year graph, except in the 2015 and 2016 corrections, where the index fell 10.97% and 10.73%. After the rally, the Dow is down 7.91% from its high.

# Azure

<https://docs.microsoft.com/en-us/learn/modules/welcome-to-azure/3-tour-of-azure-services>

Part of the introduction learning trail

Artificial Intelligence

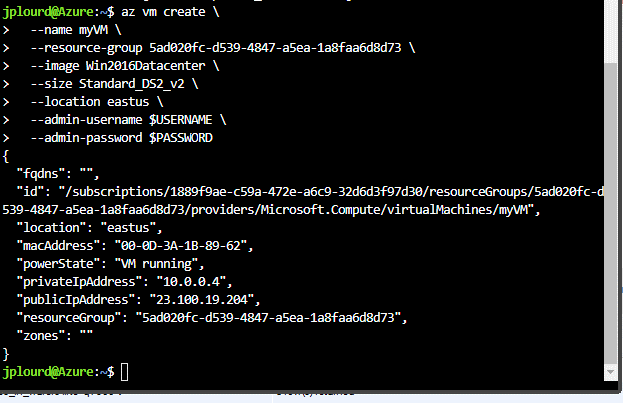
Services: Azure Machine Learning Service, Azure Machine Learning Studio

| **Service Name** | **Description** |
| --- | --- |
| Azure Machine Learning Service | Cloud-based environment you can use to develop, train, test, deploy, manage, and track machine learning models. It can auto-generate a model and auto-tune it for you. It will let you start training on your local machine, and then scale out to the cloud |
| Azure Machine Learning Studio | Collaborative, drag-and-drop visual workspace where you can build, test, and deploy machine learning solutions using pre-built machine learning algorithms and data-handling modules |

Cognitive Services

| **Service Name** | **Description** |
| --- | --- |
| Vision | Image-processing algorithms to smartly identify, caption, index, and moderate your pictures and videos. |
| Speech | Convert spoken audio into text, use voice for verification, or add speaker recognition to your app. |
| Knowledge mapping | Map complex information and data in order to solve tasks such as intelligent recommendations and semantic search. |
| Bing Search | Add Bing Search APIs to your apps and harness the ability to comb billions of webpages, images, videos, and news with a single API call. |
| Natural Language processing | Allow your apps to process natural language with pre-built scripts, evaluate sentiment and learn how to recognize what users want. |

Cloud Shell: console for interacting with Azure; browser based command line (CLI – bash or PowerShell)



"resourceGroup": "5ad020fc-d539-4847-a5ea-1a8faa6d8d73"

# Entity Framework Code First

Azure: [jplourd@outlook.com](mailto:jplourd@outlook.com) Sydney!6k

<https://docs.microsoft.com/en-us/aspnet/mvc/overview/getting-started/getting-started-with-ef-using-mvc/creating-an-entity-framework-data-model-for-an-asp-net-mvc-application>

This is the MVC 5 Tutorial

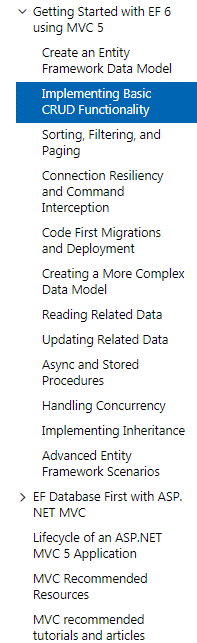
D:\Projects\back\MLNet\EntityFramework\ EntityFrameworkCodeFirst\ContosoUniversity

The downloaded completed Code First solution

D:\Projects\back\MLNet\EntityFramework\ EntityFrameworkCodeFirst\ASP.NET MVC Application Using Entity Framework Code First.zip

## TOC

A long series of turorials which gets into Razor coding, routing, etc. I’ll be jumping around to get the highlights



In this series of tutorials, you learn how to build an ASP.NET MVC 5 application that uses Entity Framework 6 for data access. This tutorial **uses the Code First workflow**. For information about how to choose between Code First, Database First, and Model First, see [Create a model](https://docs.microsoft.com/ef/ef6/modeling/).

This tutorial series explains how to build the Contoso University sample application. The sample application is a simple university website. With it, you can view and update student, course, and instructor information. Here are two of the screens you create: blah blah blah

Code first: Simple POCO data classes with EF conventions such as ‘ID’ (or classname + ‘ID’) is the primary key. Virtual ICollection<>

……This code creates a [DbSet](https://msdn.microsoft.com/library/system.data.entity.dbset(v=vs.113).aspx) property for each entity set. In Entity Framework terminology, an entity settypically corresponds to a database table, and an entity corresponds to a row in the table.

…Entity Framework can automatically create (or drop and re-create) a database for you when the application runs. You can specify that this should be done every time your application runs or only when the model is out of sync with the existing database. You can also write a Seed method that Entity Framework automatically calls after creating the database in order to populate it with test data.

The default behavior is to create a database only if it doesn't exist (and throw an exception if the model has changed and the database already exists). In this section, you'll specify that the database should be dropped and re-created whenever the model changes. Dropping the database causes the loss of all your data. This is generally okay during development, because the Seed method will run when the database is re-created and will re-create your test data. But in production you generally don't want to lose all your data every time you need to change the database schema. Later you'll see how to handle model changes by using Code First Migrations to change the database schema instead of dropping and re-creating the database.

## Conventions

The amount of code you had to write in order for Entity Framework to be able to create a complete database for you is minimal because of conventions, or assumptions that Entity Framework makes. Some of them have already been noted or were used without your being aware of them:

* The pluralized forms of entity class names are used as table names.
* Entity property names are used for column names.
* Entity properties that are named ID or classname ID are recognized as primary key properties.
* A property is interpreted as a foreign key property if it's named <navigation property name><primary key property name> (for example, StudentID for the Student navigation property since the Student entity's primary key is ID). Foreign key properties can also be named the same simply <primary key property name> (for example, EnrollmentID since the Enrollmententity's primary key is EnrollmentID).

## mdf files called ‘LocalDB’

[LocalDB](https://docs.microsoft.com/sql/database-engine/configure-windows/sql-server-2016-express-localdb?view=sql-server-2017) is a lightweight version of the SQL Server Express database engine. It's easy to install and configure, starts on demand, and runs in user mode. LocalDB runs in a special execution mode of SQL Server Express that enables you to work with databases as .mdf files. You can put LocalDB database files in the App\_Data folder of a web project if you want to be able to copy the database with the project. The user instance feature in SQL Server Express also enables you to work with .mdf files, but the user instance feature is deprecated; therefore, LocalDB is recommended for working with .mdffiles. LocalDB is installed by default with Visual Studio.

Typically, SQL Server Express is not used for production web applications. LocalDB in particular is not recommended for production use with a web application because it's not designed to work with IIS.

Now you'll create a web page to display data. The process of requesting the data automatically triggers the creation of the database. You'll begin by creating a new controller. But before you do that, build the project to make the model and context classes available to MVC controller **scaffolding**.

1. Right-click the **Controllers** folder in **Solution Explorer**, select **Add**, and then click **New Scaffolded Item**.
2. In the **Add Scaffold** dialog box, select **MVC 5 Controller with views, using Entity Framework**, and then choose **Add**.
3. Blah blah blah

The ContosoUniversity1.mdf and .ldf database files are in the %USERPROFILE% folder. C:\users\ma043651 and sure enough, it’s there.

## Implement CRUD

<https://docs.microsoft.com/en-us/aspnet/mvc/overview/getting-started/getting-started-with-ef-using-mvc/implementing-basic-crud-functionality-with-the-entity-framework-in-asp-net-mvc-application>

create,read,update,delete….

The key value is passed to the method as the id parameter and comes from route data in the **Details** hyperlink on the Index page.

### Tip: **Route data**

Route data is data that the model binder found in a URL segment specified in the routing table. For example, the default route specifies controller, action, and id segments:

routes.MapRoute(

name: "Default",

url: "{controller}/{action}/{id}",

defaults: new { controller = "Home", action = "Index", id = UrlParameter.Optional }

);

In the following URL, the default route maps Instructor as the controller, Index as the actionand 1 as the id; these are route data values.

http://localhost:1230/Instructor/Index/1**?courseID=2021**

?courseID=2021 is a query string value. The model binder will also work if you pass the id as a query string value:

http://localhost:1230/Instructor/Index?id=1&CourseID=2021

The URLs are created by ActionLink statements in the Razor view. In the following code, the idparameter matches the default route, so id is added to the route data.

@Html.ActionLink("Select", "Index", new { id = item.PersonID })

## Migrating DB

<https://docs.microsoft.com/en-us/aspnet/mvc/overview/getting-started/getting-started-with-ef-using-mvc/migrations-and-deployment-with-the-entity-framework-in-an-asp-net-mvc-application>

1. Disable the initializer that you set up earlier by commenting out or deleting the contextselement that you added to the application Web.config file.

<entityFramework>

<!--<contexts>

<context type="ContosoUniversity.DAL.SchoolContext, ContosoUniversity">

<databaseInitializer type="ContosoUniversity.DAL.SchoolInitializer, ContosoUniversity" />

</context>

</contexts>-->

<defaultConnectionFactory type="System.Data.Entity.Infrastructure.LocalDbConnectionFactory, EntityFramework">

<parameters>

<parameter value="v11.0" />

</parameters>

</defaultConnectionFactory>

<providers>

<provider invariantName="System.Data.SqlClient" type="System.Data.Entity.SqlServer.SqlProviderServices, EntityFramework.SqlServer" />

</providers>

</entityFramework>

1. Also in the application *Web.config* file, change the name of the database in the connection string to ContosoUniversity2.

<connectionStrings>

<add name="SchoolContext" connectionString="Data Source=(LocalDb)\v11.0;Initial Catalog=ContosoUniversity2;Integrated Security=SSPI;" providerName="System.Data.SqlClient" />

</connectionStrings>

This change sets up the project so that the first migration creates a new database. This isn't required but you'll see later why it's a good idea.

1. From the **Tools** menu, select **NuGet Package Manager** > **Package Manager Console**.
2. At the PM> prompt enter the following commands:

enable-migrations

add-migration InitialCreate

Where ‘InitialCreate’ is the name of the snapshot; also name of the migration class

public partial class InitialCreate : DbMigration

Run the migration, creates the C:\Users\ma043651\ ContosoUniversity2.mdf database

Connect to it through Server Explorer in Visual Studio (see below ‘MDF connection’)

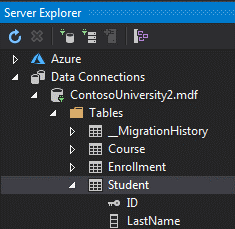
In the **Package Manager Console** window, enter the following command:

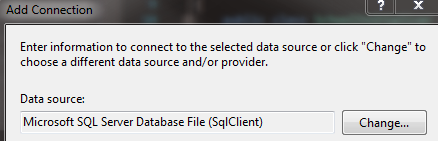
update-database

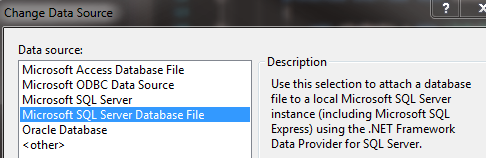
The update-database command runs the Up method to create the database and then it runs the Seed method to populate the database. The same process will run automatically in production after you deploy the application, as you'll see in the following section.

### MDF connection

Server Explorer in VS. Right click ‘Data Connections’ -> add -> Database file -> database file -> browse to .mdf -> Bob’s ur uncle







## Attributes

public string FirstMidName { get; set; }

[**DataType**(DataType.Date)]

[**DisplayFormat**(DataFormatString = "{0:yyyy-MM-dd}", ApplyFormatInEditMode = true)]

public DateTime EnrollmentDate { get; set; }

The ***Column*** attribute specifies that when the database is created, the column of the Student table that maps to the FirstMidName property will be named FirstName. In other words, when your code refers to Student.FirstMidName, the data will come from or be updated in the FirstName column of the Student table. If you don't specify column names, they are given the same name as the property name.

[Column("FirstName")]

public string FirstMidName { get; set; }

[Key]

[**ForeignKey**("Instructor")]

public int InstructorID { get; set; }

You can also use the Key attribute if the entity does have its own primary key but you want to name the property something different than classnameID or ID. By default EF treats the key as non-database-generated because the column is for an identifying relationship.

### The ForeignKey Attribute

When there is a one-to-zero-or-one relationship or a one-to-one relationship between two entities (such as between OfficeAssignment and Instructor), EF can't work out which end of the relationship is the principal and which end is dependent.   
The [ForeignKey Attribute](https://msdn.microsoft.com/library/system.componentmodel.dataannotations.schema.foreignkeyattribute.aspx) can be applied to the dependent class to establish the relationship. If you omit the [ForeignKey Attribute](https://msdn.microsoft.com/library/system.componentmodel.dataannotations.schema.foreignkeyattribute.aspx), you get the following error when you try to create the migration:

Unable to determine the principal end of an association between the types ………..

Altered the entities in code, now create a new migration

PM> add-Migration ComplexDataModel

If the DB (file) contains data then the foreign key dependencies have to be fixed up.

Edit the file Migrations\timestamp\_ComplexDataModel

CreateTable(

"dbo.CourseInstructor",

c => new

{

CourseID = c.Int(nullable: false),

InstructorID = c.Int(nullable: false),

})

.PrimaryKey(t => new { t.CourseID, t.InstructorID })

.ForeignKey("dbo.Course", t => t.CourseID, cascadeDelete: true)

.ForeignKey("dbo.Instructor", t => t.InstructorID, cascadeDelete: true)

.Index(t => t.CourseID)

.Index(t => t.InstructorID);

// Create a department for course to point to.

Sql("INSERT INTO dbo.Department (Name, Budget, StartDate) VALUES ('Temp', 0.00, GETDATE())");

// default value for FK points to department created above.

AddColumn("dbo.Course", "DepartmentID", c => c.Int(nullable: false, defaultValue: 1));

//AddColumn("dbo.Course", "DepartmentID", c => c.Int(nullable: false));

AlterColumn("dbo.Course", "Title", c => c.String(maxLength: 50));

## Class inheritance in DF

<https://docs.microsoft.com/en-us/aspnet/mvc/overview/getting-started/getting-started-with-ef-using-mvc/implementing-inheritance-with-the-entity-framework-in-an-asp-net-mvc-application>

If student and instructor classes changed to inherit from ‘Person’ class containing the common elements of firstname lastname, how is it represented in EF?

## SQL Server Management Studio SSMS

……….it would run it on Master database and not on our database. Just for convenience, you can **set up the default DB** as so:

1. In your object explorer > Expand Security > Expand Logins  
2. Right click your login (in this case WeiL) and select ‘Properties’  
3. Under General, you will see a drop down called “Default Database”  
4. And ta-da! its done.

SSMS scripts .sql files are stored here:

C:\Users\ma043651\OneDrive - M A-COM Technology Solutions, Inc\Documents\SQL Server Management Studio

<https://docs.microsoft.com/en-us/aspnet/mvc/overview/getting-started/getting-started-with-ef-using-mvc/connection-resiliency-and-command-interception-with-the-entity-framework-in-an-asp-net-mvc-application>

In this tutorial you'll learn how to use connection resiliency and command interception. They are two important features of Entity Framework 6 that are especially valuable when you are deploying to the cloud environment: connection resiliency (automatic retries for transient errors) and command interception (catch all SQL queries sent to the database in order to log or change them).

## SQL Server Connection Strings for ASP.NET Web Applications

<https://docs.microsoft.com/en-us/previous-versions/aspnet/jj653752(v=vs.110)>

# Entity Framework Database First

<https://docs.microsoft.com/en-us/aspnet/mvc/overview/getting-started/database-first-development/>

A long series of turorials

## TOC

