1. In the sense of machine learning, what is a model? What is the best way to train a model?

A machine learning model is a file that has been trained to recognize certain types of patterns. You train a model over a set of data, providing it an algorithm that it.

Training a model simply means learning (determining) good values for all the weights and the bias from labeled examples. In supervised learning, a machine learning algorithm builds a model by examining many examples and attempting to find a model that minimizes loss; this process is called empirical risk minimization.

Loss is the penalty for a bad prediction. That is, loss is a number indicating how bad the model's prediction was on a single example. If the model's prediction is perfect, the loss is zero; otherwise, the loss is greater. The goal of training a model is to find a set of weights and biases that have low loss, on average, across all examples.

2. In the sense of machine learning, explain the "No Free Lunch" theorem.

The "No Free Lunch" theorem states that when averaged over all optimization problems without resampling, all optimization strategies perform equally well. The most significant effects on optimization, search and supervised learning have come from this basic theoretical idea.

3. Describe the K-fold cross-validation mechanism in detail.

In machine learning (ML), generalization usually refers to the ability of an algorithm to be effective across various inputs. It means that the ML model does not encounter performance degradation on the new inputs from the same distribution of the training data.

For human beings generalization is the most natural thing possible. We can classify on the fly. For example, we would definitely recognize a dog even if we didn’t see this breed before. Nevertheless, it might be quite a challenge for an ML model. That’s why checking the algorithm’s ability to generalize is an important task that requires a lot of attention when building the model.

To do that, we use Cross-Validation (CV).

Cross-validation is a technique for evaluating a machine learning model and testing its performance. CV is commonly used in applied ML tasks. It helps to compare and select an appropriate model for the specific predictive modeling problem.

CV is easy to understand, easy to implement, and it tends to have a lower bias than other methods used to count the model’s efficiency scores. All this makes cross-validation a powerful tool for selecting the best model for the specific task.

There are a lot of different techniques that may be used to cross-validate a model. Still, all of them have a similar algorithm:

Divide the dataset into two parts: one for training, other for testing

Train the model on the training set

Validate the model on the test set

Repeat 1-3 steps a couple of times. This number depends on the CV method that you are using

As you may know, there are plenty of CV techniques. Some of them are commonly used, others work only in theory. Let’s see the cross-validation methods that will be covered in this article.

* Hold-out
* K-folds
* Leave-one-out
* Leave-p-out
* Stratified K-folds
* Repeated K-folds
* Nested K-folds
* Time series CV

Hold-out cross-validation

Hold-out cross-validation is the simplest and most common technique. You might not know that it is a hold-out method but you certainly use it every day.

The algorithm of hold-out technique:

* Divide the dataset into two parts: the training set and the test set. Usually, 80% of the dataset goes to the training set and 20% to the test set but you may choose any splitting that suits you better
* Train the model on the training set
* Validate on the test set
* Save the result of the validation

import numpy as np

from sklearn.model\_selection import train\_test\_split

X, y = np.arange(10).reshape((5, 2)), range(5)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y,

test\_size=0.2,

random\_state=111)

k-Fold cross-validation

k-Fold cross-validation is a technique that minimizes the disadvantages of the hold-out method. k-Fold introduces a new way of splitting the dataset which helps to overcome the “test only once bottleneck”.

The algorithm of the k-Fold technique:

* Pick a number of folds – k. Usually, k is 5 or 10 but you can choose any number which is less than the dataset’s length.
* Split the dataset into k equal (if possible) parts (they are called folds)
* Choose k – 1 folds as the training set. The remaining fold will be the test set
* Train the model on the training set. On each iteration of cross-validation, you must train a new model independently of the model trained on the previous iteration
* Validate on the test set
* Save the result of the validation
* Repeat steps 3 – 6 k times. Each time use the remaining fold as the test set. In the end, you should have validated the model on every fold that you have.
* To get the final score average the results that you got on step 6.

import numpy as np

from sklearn.model\_selection import KFold

X = np.array([[1, 2], [3, 4], [1, 2], [3, 4]])

y = np.array([1, 2, 3, 4])

kf = KFold(n\_splits=2)

for train\_index, test\_index in kf.split(X):

print("TRAIN:", train\_index, "TEST:", test\_index)

X\_train, X\_test = X[train\_index], X[test\_index]

y\_train, y\_test = y[train\_index], y[test\_index]

n general, it is always better to use k-Fold technique instead of hold-out. In a head to head, comparison k-Fold gives a more stable and trustworthy result since training and testing is performed on several different parts of the dataset. We can make the overall score even more robust if we increase the number of folds to test the model on many different sub-datasets.

Still, k-Fold method has a disadvantage. Increasing k results in training more models and the training process might be really expensive and time-consuming.

Leave-one-out cross-validation

Leave-one-out сross-validation (LOOCV) is an extreme case of k-Fold CV. Imagine if k is equal to n where n is the number of samples in the dataset. Such k-Fold case is equivalent to Leave-one-out technique.

The algorithm of LOOCV technique:

* Choose one sample from the dataset which will be the test set
* The remaining n – 1 samples will be the training set
* Train the model on the training set. On each iteration, a new model must be trained
* Validate on the test set
* Save the result of the validation
* Repeat steps 1 – 5 n times as for n samples we have n different training and test sets
* To get the final score average the results that you got on step 5.

import numpy as np

from sklearn.model\_selection import LeaveOneOut

X = np.array([[1, 2], [3, 4]])

y = np.array([1, 2])

loo = LeaveOneOut()

for train\_index, test\_index in loo.split(X):

print("TRAIN:", train\_index, "TEST:", test\_index)

X\_train, X\_test = X[train\_index], X[test\_index]

y\_train, y\_test = y[train\_index], y[test\_index]

4. Describe the bootstrap sampling method. What is the aim of it?

Bootstrap sampling is used in a machine learning ensemble algorithm called bootstrap aggregating (also called bagging). It helps in avoiding overfitting and improves the stability of machine learning algorithms. In bagging, a certain number of equally sized subsets of a dataset are extracted with replacement.

Bootstrap Sampling: It is a method in which we take a sample data repeatedly with replacement from a data set to estimate a population parameter. It is used to determine various parameters of a population.

5. What is the significance of calculating the Kappa value for a classification model? Demonstrate how to measure the Kappa value of a classification model using a sample collection of results.

Kappa can range from 0 to 1. A value of 0 means that there is no agreement between the raters (real-world observer vs classification model), and a value of 1 means that there is perfect agreement between the raters. In most cases, anything over 0.7 is considered to be very good agreement.

6. Describe the model ensemble method. In machine learning, what part does it play?

Ensemble methods are techniques that aim at improving the accuracy of results in models by combining multiple models instead of using a single model. The combined models increase the accuracy of the results significantly. This has boosted the popularity of ensemble methods in machine learning

7. What is a descriptive model's main purpose? Give examples of real-world problems that descriptive models were used to solve.

A descriptive model describes a system or other entity and its relationship to its environment. It is generally used to help specify and/or understand what the system is, what it does, and how it does it. A geometric model or spatial model is a descriptive model that represents geometric and/or spatial relationships.

8. Describe how to evaluate a linear regression model.

How to evaluate regression model , let’s start

1. R Square/Adjusted R Square

2. Mean Square Error(MSE)/Root Mean Square Error(RMSE)

3. Mean Absolute Error(MAE)

4. illustrate Residual of model as a normal distribution ( bell shape)

5. By OLS from statemodels.formula

R Square/Adjusted R Square :

This is a first measure of regression model especially we, everybody, do during evaluation because it is easy to interpret score between 0 to 1. If we see good score like close to 1, then we assume that model is good fit. Of course , R Square is a good measure to determine how well the model fits the dependent variables. However, it does not take into consideration of overfitting problem. If your regression model has many independent variables, because the model is too complicated, it may fit very well to the training data but performs badly for testing data.So I recommend that we have to see all perspective for better evaluation . let’s talk what is actually mean R² . R² is calculated by the sum of squared of prediction error divided by the total sum of square which replace the calculated prediction with mean. R Square value is between 0 to 1 and bigger value indicates a better fit between prediction and actual value.

Mean Square Error(MSE)/Root Mean Square Error(RMSE):

while R² is a relative measure of how well the model fit dependent variables, whereas Mean Square Error is an absolute measure of the fit of model. MSE is calculated by sum of square of prediction error. Where prediction error is minus between true values and prediction values, and then it is made by square because we avoid negative error score. It’s result gives us how much deviation from actual number. It’s number might be larger number which may be like uncommon .

Explore Residual :

At last for evaluation , We can also explore residuals, which comes from true values and predicted values, by scatterplot or diskplot of searbearn library or matplotlib. If we get linear shape on scatter plot or bell shape in distplot , then we can pretty say that model fit perfectly, and can predict very close to real values.

OLS from statemodels.formula:

By statemodels library , we can explore all over summary on one place like R² , R² adjust , coeff etc…

## Evaluation Regression model by all metrics

## Data is taken from kaggle which dummy data only for practice.

# import necessaries libraries

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

%matplotlib inline

import seaborn as sns

# for warning message

import warnings

warnings.filterwarnings("ignore")

# import data

df\_original = pd.read\_csv("Ecommerce Customers.csv")

df = df\_original.copy()

df.head()

# check datatype of features and null values

df.info()

# check statistics overall

df.describe()

# data Exploration

sns.pairplot(df)

# check correlation

sns.heatmap(df.corr(),annot= True)

# split data

X = df[['Avg. Session Length','Time on App','Time on Website','Length of Membership']]

X.head()

y = df['Yearly Amount Spent']

y

# mean values

np.mean(df['Yearly Amount Spent'])

# import class , methods from sklearn libraries

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LinearRegression

# split data for training and test set data

X\_train,X\_test,y\_train,y\_test = train\_test\_split(X,y,test\_size = 0.2,random\_state = 42)

# check dimension of every splited dataset

print(df.shape)

print(X\_train.shape)

print(X\_test.shape)

print(y\_train.shape)

print(y\_test.shape)

# fit data in linearRegression

lg = LinearRegression()

lg.fit(X\_train,y\_train)

lg.score(X\_train,y\_train)

#0.9854240629700333

# predit value

y\_pred = lg.predict(X\_test)

# import evaluation metrics from sklearn library

from sklearn.metrics import r2\_score,mean\_squared\_error,mean\_absolute\_error

# R2 score which lies between 0 to 1. close to 1 score shows better a fit model

print("R2 Score")

r2\_score(y\_pred,y\_test)

#0.9782625350414402

# mean square error

print("Mean Square Error")

mean\_squared\_error(y\_test,y\_pred)

#109.86374118394116

print("Root Mean Square Error")

print(np.sqrt(mean\_squared\_error(y\_test,y\_pred)))

#10.48159058463653

# Mean Absolute Error

print("Mean Absolute Error")

mean\_absolute\_error(y\_test,y\_pred)

#8.558441885315286

# explore residual

residual = y\_test - y\_pred

sns.distplot(residual)

sns.scatterplot(y\_test,y\_pred)

plt.scatter(y\_test,y\_test)

plt.xlabel("Real Values")

plt.ylabel("predicted values")

# Coeffecient of model

coeffecients = pd.DataFrame(lg.coef\_,X.columns)

coeffecients.head()

# Avg. Session Length 25.596259

# Time on App 38.785346

# Time on Website 0.310386

# Length of Membership 61.896829

# evaluate from statemodels

import statsmodels.formula.api as smf

# merge data X\_train and y\_train for ols formula

train\_data = pd.merge(y\_train, X\_train, left\_index=True, right\_index=True)

# remove all white space from columns name by rename

train\_data.rename(columns={'Yearly Amount Spent':'Yearly\_Spent', 'Avg. Session Length':'Avg\_SessionLeg', 'Time on App':'Time\_App',

'Time on Website':'Time\_Website', 'Length of Membership':'Length\_Mem'},inplace= True)

df\_dependent = train\_data['Yearly\_Spent']

df\_independent = train\_data.drop(labels = ['Yearly\_Spent'],axis =1)

# making formula

featureFormula = "+".join(df\_independent.columns)

sm\_formula = "Yearly\_Spent ~ " + featureFormula

# fit model

results = smf.ols(sm\_formula, data=train\_data).fit()

results.summary()

9. Distinguish :

1. Descriptive vs. predictive models

Models that are primarily used for understanding, predicting and communicating are referred to as descriptive models, whereas models mainly used for implementation are called prescriptive models. This contribution focuses on teaching both the common and the distinguishing aspects of the two model categories.

Predictive and prescriptive analytics inform your business strategies based on collected data. Predictive analytics forecasts potential future outcomes, while prescriptive analytics helps you draw specific recommendations.

2. Underfitting vs. overfitting the model

Underfitting means that your model makes accurate, but initially incorrect predictions. In this case, train error is large and val/test error is large too. Overfitting means that your model makes not accurate predictions. In this case, train error is very small and val/test error is large.

3. Bootstrapping vs. cross-validation

Cross validation splits the available dataset to create multiple datasets, and Bootstrapping method uses the original dataset to create multiple datasets after resampling with replacement. Bootstrapping it is not as strong as Cross validation when it is used for model validation.

10. Make quick notes on:

1. LOOCV.

The Leave-One-Out Cross-Validation, or LOOCV, procedure is used to estimate the performance of machine learning algorithms when they are used to make predictions on data not used to train the model.

2. F-measurement

An F-score is the harmonic mean of a system's precision and recall values. It can be calculated by the following formula: 2 x [(Precision x Recall) / (Precision + Recall)].

That is, a good F1 score means that you have low false positives and low false negatives, so you're correctly identifying real threats and you are not disturbed by false alarms. An F1 score is considered perfect when it's 1 , while the model is a total failure when it's 0

he F-score, also called the F1-score, is a measure of a model's accuracy on a dataset. It is used to evaluate binary classification systems, which classify examples into 'positive' or 'negative'.

3. The width of the silhouette

Silhouette width is a widely used index for assessing the fit of individual objects in the classification, as well as the quality of clusters and the entire classification.

The silhouette value is a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation). The silhouette ranges from −1 to +1, where a high value indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters.

4. Receiver operating characteristic curve

An ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters: True Positive Rate. False Positive Rate.

The ROC curve illustrates the relationship between sensitivity and FPR. Because the ROC curve displays the sensitivities and FPRs at all possible cutoff levels, it can be used to assess the performance of a test independently of the decision threshold

The primary method used for this process is the receiver operating characteristic (ROC) curve. The ROC curve is used to assess the overall diagnostic performance of a test and to compare the performance of two or more diagnostic tests.

AREA UNDER THE ROC CURVE

In general, an AUC of 0.5 suggests no discrimination (i.e., ability to diagnose patients with and without the disease or condition based on the test), 0.7 to 0.8 is considered acceptable, 0.8 to 0.9 is considered excellent, and more than 0.9 is considered outstanding.