School of Electrical and Computer Engineering- University of Tehran

Artificial Intelligence Course- 2024

Assignment 2-Due Date: May 23, 2024, before 4 PM

[1].This assignment has four parts. For this assignment, I have provided all the codes and elucidations for the K-Nearest Neighbors (KNN), Support Vector Machines(SVM), Gradient Boosted Tree (GBT), and Extreme Gradient Boosting(XGBoost). Images/slides are provided in the [Assignment 2-Due Date: May 23, 2024, before 4 PM] folder. Also, three datasets are provided. You need to upload them to run the notebook. You must provide numerical (manual) examples and explain all these machine-learning algorithms similar to the examples I provided in the lecture. Your examples must be clear. Give one numerical example for each algorithm and demonstrate how each algorithm works. This is very important. Remember, we learn by examples. Indeed, deep neural networks learn by examples.

[2]. You must insert your slides into the notebook that I have provided. The name of the notebook is [Assignment 2-Notebook[KNN-SVM-GBT-XGBoost]], and it is in the [Assignment 2-Due Date: May 23, 2024, before 4 PM] folder. Furthermore, you must submit the slides that you insert in the notebook.

[3]. You must also explain the codes for all four algorithms.

[4]. You must record each part of your work and submit only one video. [The length of your recording for each part should be $\leq$ 15 mins].

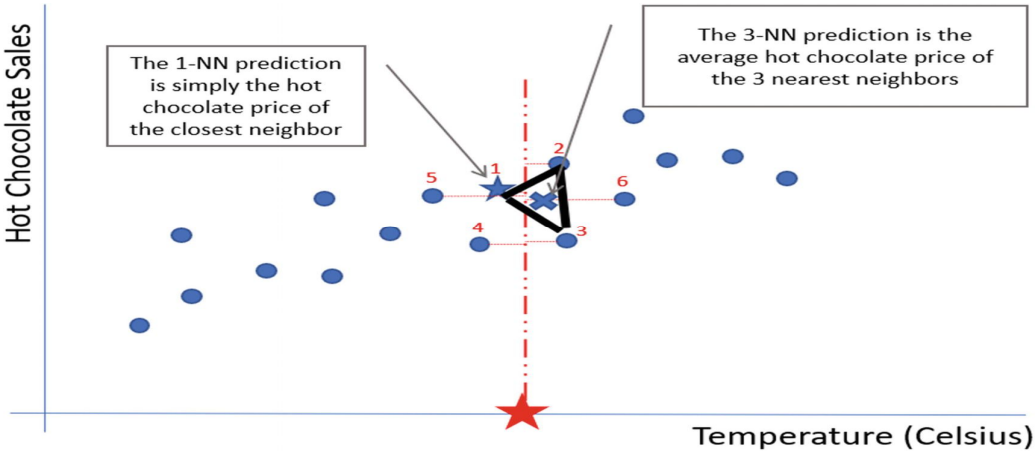
Note: For Support Vector Machine, you must provide an example for Radial Basis Function(RBF) Kernel

Part 1-The K-Nearest Neighbors (KNN) Algorithm

The definition of nearest neighbors is based on the computation of the Euclidean distance from the new data point to each of the existing data and is given by

$$d(p,q)=\sqrt{(p_1-q_1)^2+(p_2-q_2)^2+...(p_n-q_n)^2}$$

When we use the KNN algorithm, we need to identify the closest neighbors. We will not have a prediction once we have identified the neighbors closest to our new data point. There is one step remaining to convert the multiple neighbors into one prediction. There are two prevalent methods for it. The first method is to take the average of the target value of the k nearest neighbors. This average is then used as the prediction. The second method is to take the weighted average of the k nearest neighbors and use their distances as the inverse weight so that closer points are weighted heavier in the prediction. Now, how many nearest neighbors should we include in the prediction? The value of k decides this. To apply this, let us see two cases – one nearest neighbor and three nearest neighbors – and see the difference in prediction. The two are given in the figure below.



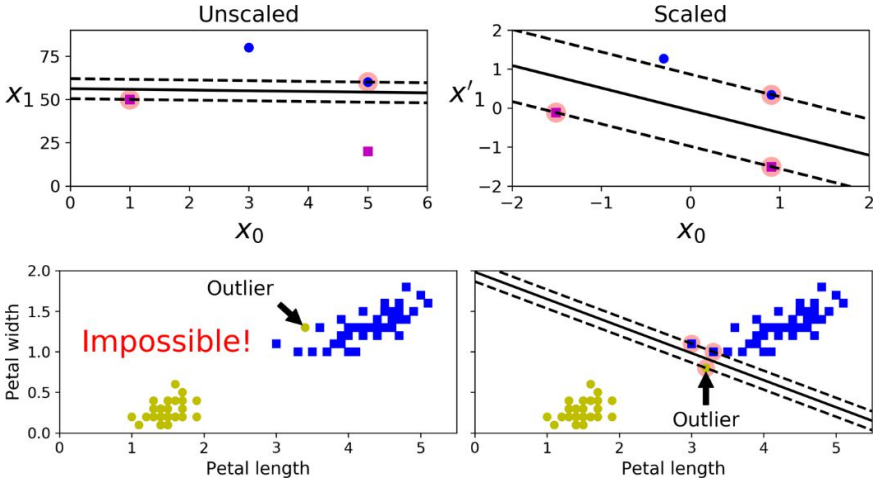
Part II-Support Vector Machines Algorithm

[A]. The fundamental idea behind Support Vector Machines is to fit the largest possible street/margin between the decision boundary that separates the two classes and the training instances. When performing soft margin classification, the SVM searches for a compromise between perfectly separating the two classes and having the widest possible Margin. Another key idea is to use kernels when training on nonlinear datasets.

[B]. The support vectors entirely determine the decision boundary. Any instance that is not a support vector (i.e., is off the street/margin) has no influence whatsoever; you could remove them, add more instances, or move them around, and as long as they stay off the street, they will not affect the decision boundary. Computing the predictions only involves the support vectors, not the whole training set.

[C].SVMs will fit the largest possible "street" between the classes, so if the training set is not scaled, the SVM will tend to neglect small features.

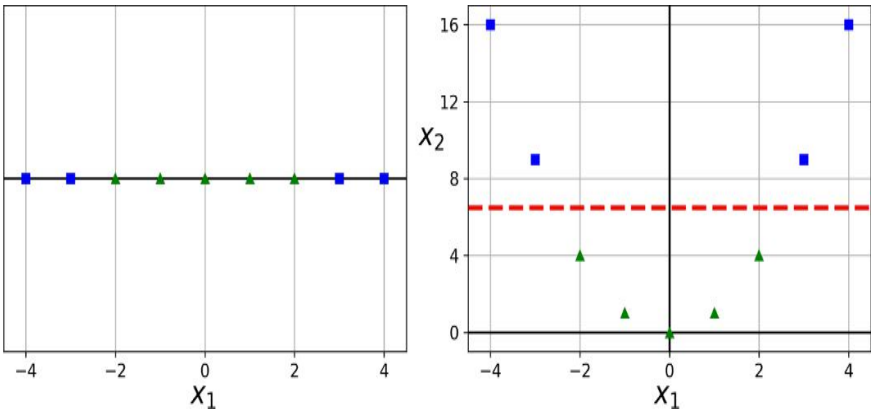
[D]. The SVM classifier can output the distance between the test instance and the decision boundary. We can use this as a confidence score. However, this score cannot be directly converted into an estimation of the class probability. If you set probability=True when creating an SVM in Scikit-Learn, then after training, it will calibrate the probabilities using Logistic Regression on the SVM's scores (trained by an additional five-fold cross-validation on the training data). This will add the predict_proba() and predict_log_proba() methods to the SVM. See the figures below.



[E]. Soft Margin Classification-If we strictly impose that all instances must be off the street and on the right side, this is called hard margin classification. There are two main issues with hard margin

classification. First, it only works if the data is linearly separable. Second, it is sensitive to outliers. To avoid these issues, we use a more flexible model. The objective is to find a good balance between keeping the street as large as possible and limiting the margin violations (i.e., instances that end up in the middle of the street or even on the wrong side). This is called soft margin classification.

[F]. Nonlinear SVM Classification- Although linear SVM classifiers are efficient and work well in many cases, and many datasets are not even close to being linearly separable. One approach to handling nonlinear datasets is to add more features, such as polynomial features. In some cases, this can result in a linearly separable dataset. Consider the plot below, representing a simple dataset with just one feature x_1. This dataset is not linearly separable. However, adding a second feature · makes the resulting 2D dataset perfectly linearly separable.



Part III-The Gradient-Boosting Algorithm

Gradient Boosted Tree (GBT) is another tree-based ensemble algorithm similar to Random Forest. GBTs use a technique known as boosting to create a strong learner from weak learners (shallow trees). GBTs train an ensemble of decision trees sequentially, with each succeeding tree decreasing the error of the previous tree. This is done by using the residuals of the previous model to fit the next model. This residual-correction process is performed a set number of iterations, with the number of iterations determined by cross-validation until the residuals have been fully minimized.

Gradient-Boosted Trees Example

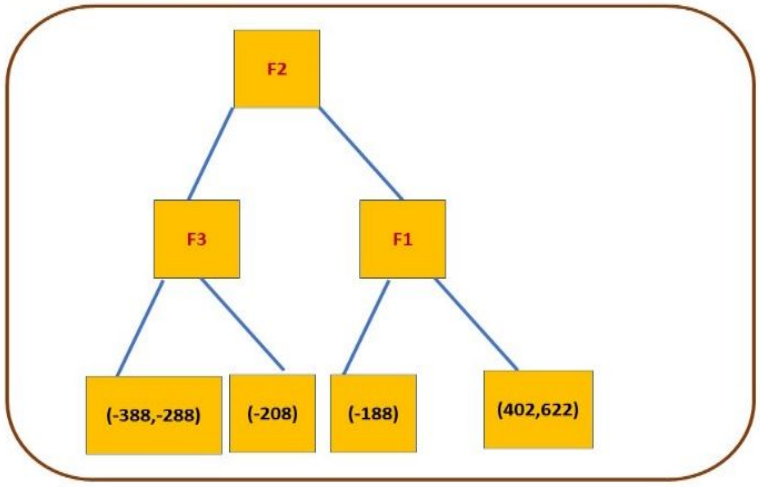
Feature 1	Feature 2	Feature 3	Output
5	1500	5	480
11	2030	12	1090
14	1442	6	350
8	2501	4	1310
12	1300	9	400
10	1789	11	500

1. Calculate the average of the target label= $480+$1090+$350+$1310+$400+500=$688

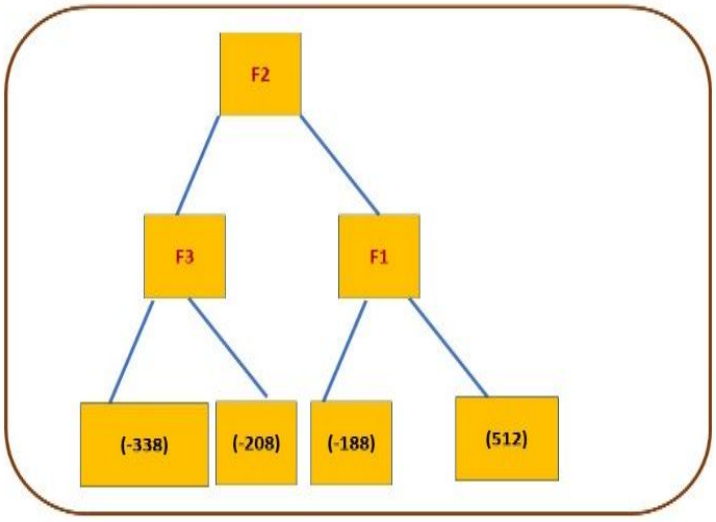
2. Calculate the residuals

Feature 1	Feature 2	Feature 3	Output	Residuals= Actual Value – Predicted Value
5	1500	5	480	-208
11	2030	12	1090	402
14	1442	6	350	-338
8	2501	4	1310	622
12	1300	9	400	-288
10	1789	11	500	-188

3. Construct a decision tree: We build a tree to predict the residuals(not the desired label).



4. Since there are more residuals than leaves, we compute their average and place that inside the leaf.



5. We Predict the target label using all of the trees within the ensemble.

Predicted Output = Average Output + Learning Rate (Residuals Predicted by the Decision Tree)

Predicted Output =688 + 0.1(-338) =654.2

6. We compute the new residuals

Old Residuals	New Residuals
-208	-187.2
402	350.8
-338	-304.8
622	570.8
-288	-254.2
-188	-169.2

7. We repeat steps 3 to 5 until the number of iterations matches the number of estimators.

8.After we train our model, we use all the trees in our ensemble to make a final prediction

Part IV-Extreme Gradient Boosting(XGBoost)Algorithm

XGBoost is one of the best gradient-boosted tree implementations currently available. Released on March 27, 2014, by Tianqi Chen as a research project, XGBoost has become the dominant machine learning algorithm for classification and regression. XGBoost was designed using the general principles of gradient boosting, combining weak learners into a strong learner. But while gradient-boosted trees are built sequentially – slowly learning from data to improve its prediction in succeeding iterations, XGBoost builds trees in parallel.XGBoost produces better prediction performance by controlling model complexity and reducing overfitting through its built-in regularization. XGBoost uses an approximate algorithm to find split points when finding the best-split points for a continuous feature. The approximate splitting method uses discrete bins to bucket continuous features, significantly speeding up model training. XGBoost includes another tree-growing method using a histogram-based algorithm, providing an even more efficient method of bucketing continuous features into discrete bins. However, while the approximate method creates a new set of bins per iteration, the histogram-based approach reuses bins over multiple iterations. This approach allows for additional optimizations that are not achievable with the approximate method, such as the ability to cache bins and parent and sibling histogram subtraction. To optimize sorting operations, XGBoost stores sorted data in in-memory units of blocks. Sorting blocks can be efficiently distributed and performed by parallel CPU cores. XGBoost can effectively handle weighted data via its weighted quantile sketch algorithm, can efficiently handle sparse data, is cache-aware, and supports out-of-core computing by utilizing disk space for large datasets, so data does not have to fit in memory.

Important Note

1. For this assignment, you should record a video. You should show your face. Your voice must be clear. It is your responsibility to make sure your video is working. You should put your video and notebook in Google Drive. Your folder should have your[student ID number, first name, and last name]. Only one video should be submitted for all parts of this assignment.
2. Your notebook should be well structured, and before each code block, you should explain the codes of the next block. You should follow the format of the notebooks I presented in the lectures.
3. You should make your Google link open access and submit your video to the auxiliary TA of our course, Mr Behzad Mohasel Afshari, via email on May 23, 2024, before 4 PM. His email for this course is ai.2024.cs@gmail.com.
4. You should use Google Colab. Insert all the slides into your Google Colab environment, then record all parts of this assignment.
5. You should submit the following two items
A. Your Google Colab notebook with additional slides.
B. Your video file extension should be .mp4. Other formats will not be accepted.
6. Your video file and notebook should be saved as [student ID number, first name, and last name]. Your total video file should not be more than 45 minutes. Your recording must be in English.
7. You should ensure that your codes and mathematical algorithms are flawless.

If further elucidation is warranted, please don't hesitate to contact me.