A PROJECT REPORT

ON

“HOUSE PRICE PREDICTION” using TensorFlow.

For ‘Advanced Programming’ Course

BY

TOKTASYN ALMIRA

IT-2107

SUPERVISED BY

SULTANMURAT YELEU

Senior-lecturer

BSc IN COMPUTER SCIENCE, 2nd YEAR

ASTANA IT UNIVERSITY, ASTANA, KAZAKHSTAN

2023

YT-VIDEO: https://youtu.be/ozzoCG8CbXE

GITHUB LINK: https://github.com/Almmira/final

1. Introduction

1.1 Problem

Initially, I was in a couple, but we decided to do each a separate project.

My topic is the prediction of housing prices, which involves the development of a model for forecasting housing prices. This model is crucial for providing important information and enhancing the efficiency of the real estate market. The main objective of my project is to assist individuals in finding their ideal home, which is a future home. As housing prices continue to escalate each year, there is a need for a reliable mechanism that can accurately forecast future housing prices. Real estate appraisers, landowners, and others can use the housing price forecasting model to estimate the value of a house and determine a suitable sale price. This will enable prospective buyers to make an informed decision regarding the best time to purchase a house. Although the physical attributes, style, and location are the primary factors that determine the price of a house, there are various individual factors that can also affect its price.

1.2 Literature review with links (another solutions)

1) Alan Ihre & Isak Engstrom. Predicting house prices with machine learning methods. (2019)

https://www.diva-portal.org/smash/get/diva2:1354741/FULLTEXT01.pdf

In this study, the machine learning algorithms k-Nearest-Neighbours regression (k-NN) and Random Forest (RF) regression were used to predict house prices from a set of features in the Ames housing data set. The Random Forest was found to consistently perform better than the kNN algorithm in terms of smaller. errors and be better suited as a prediction model for the house price problem. With a mean absolute error of about 9 % from the mean price in the best case, the practical usefulness of the prediction is rather limited to making basic valuations.

2) House price prediction using The Ames Housing dataset. (2022)

https://github.com/sharmasapna/house-price-prediction

The aim of the project is to predict house prices for houses in the Boston Housing Dataset. Two files, train and test are provided, and the price of the test data is to be estimated. Here I have used XGBoost for prediction. They got a slight improvement with hyperparameter tuning (from 0.14065 to 0.14036).

3) Shreyas Raghavan. Create a model to predict house prices using Python. (2017)

https://github.com/Shreyas3108/house-price-prediction

Predicting house prices using Linear Regression and Gradient Boosting Regressor (GBR). Achieved accuracy is 91.94%.

1.3 Current work (description of the work)

In my project, I used my own dataset, which I got by scraping the site using BeautifulSoup. It took me a couple of days. Since the output was not quite correct for use in training the model, I redid it manually, it also took a couple of days, as my eyes got tired quickly. An additional problem was that my laptop could not withstand a load of 5,000 documents, so I reduced the amount of data. I decided to focus only on the city of Astana, as it was interesting to find out about prices here. After watching a lot of training videos and websites with additional information, I started creating the project. I uploaded the data as a CSV file.

Google Colab was used for the project, as well as libraries: TensorFlow, Keras, Pandas, and Matplotlib. My model consists of 4 layers, I also used RELU as an activation function. My model is compiled using the mean square error (MSE) as a loss function, which is a common loss function for regression problems. For optimization, I used "Adam" - an optimization algorithm. Before training, I normalized the data, for training I used 0.1 percent of the data, which allowed me to improve accuracy with a small dataset. To prevent overfitting, I used EarlyStopping, which tracked the loss of validation ('val_loss') during training and stopped the learning process if the loss of validation did not improve for 6 consecutive periods. I used 1000 epochs. I chose all these numbers because they gave the best result based on my dataset.

2. Data and Methods

2.1 Information about the data (probably analysis of the data with some visualizations)

For the project, I used the BeautifulSoup library (Python). The data was taken from the website Krisha.kz — this is the largest website of ads for the sale and rental of apartments, houses, and other real estates in Kazakhstan. I chose locally the data only about real estate in Astana because I was wondering what prices are here. The amount of data is 2.2k.

```
In [ ]: # get urls from one page
import urllib3
import json
from bs4 import BeautifulSoup

http = urllib3.PoolManager()


url = "https://krisha.kz/prodazha/kvartiry/astana/?das[house.year][to]=2022"
response = http.request('GET', url)
print(response.status)
soup = BeautifulSoup(response.data)

def get_urls(url):
    response = http.request('GET', url)
    print(response.status)
    soup = BeautifulSoup(response.data)
    links = []
    for i in soup.find_all("a", {"class": "a-card__title"}):
        links.append(i.get("href"))
    return links
```

```
In [ ]: url = "https://krisha.kz/prodazha/kvartiry/astana/?das[house.year][to]=2022&page="
links = []

import random
import time
random.randint(0,9)
count = 1
for i in range(1, 301):
    response = get_urls(url + str(i))
    print(count, "page", end=",  ")
    count += 1
    links.extend(response)
    time.sleep(random.randint(2,5))
```

```
In [ ]: def get_info(url):
    response = http.request('GET', url)
    if response.status != 200:
        d = {"error": response.status}
        return d
    soup = BeautifulSoup(response.data)
    offerShort = soup.find_all("div", {"offer__short-description"})

    # get offer_info
    titles = []
    for i in offerShort[0].find_all("div", {"class": "offer__info-title"}):
        titles.append(i.get_text())

    values = []
    for i in offerShort[0].find_all("div", {"class": "offer__advert-short-info"}):
        values.append(" ".join(i.get_text().split()))

    d =  dict(zip(titles, values))

    # get parameters

    offerParams = soup.find("div", {"offer__parameters"})
    paramKeys = []
    for i in offerParams.find_all("dt"):
        paramKeys.append(i.get_text())
    values = []
    for i in offerParams.find_all("dd"):
        values.append(i.get_text())

    d1 = dict(zip(paramKeys, values))

    # get jsdata
    js = soup.find("script", {"id": "jsdata"})
    txt = js.get_text()
    d2 = json.loads(txt[txt.find("{"):txt.rfind("}")+1])
    d2 = d2["advert"]
    d2.pop("photos")
    item = {}
    item["offer"] = d
    item["params"] = d1
    item["data"] = d2

    return item
```

```
In [ ]: data = []
count = 1
for i in links:
    url = "https://krisha.kz" + i
    print(count, end=" ")
    count += 1
    item = get_info(url)
    data.append(item)
    time.sleep(random.randint(2,5))
```

```
In [ ]: links = list(set(links))
```

```
In [ ]: len(links)
```

```
In [ ]: data[1]
```

```
In [ ]: response.status != 200
```

```
In [ ]: len(data)
```

```
In [ ]: toExcel = sorted(data, key = lambda x: len(x))[8:]
```

```
In [ ]: sorted(toExcel, key = lambda x: len(x["params"]))[-1]
```

```
In [ ]: toExcel[0]["offer"]
```

```
In [ ]: Fruit_Json = json.dumps(toExcel)
print(Fruit_Json)
```

```
In [ ]: with open('all.json'.format(1), 'w', encoding='utf-8') as file:
    json.dump(data, file, ensure_ascii=False)
```

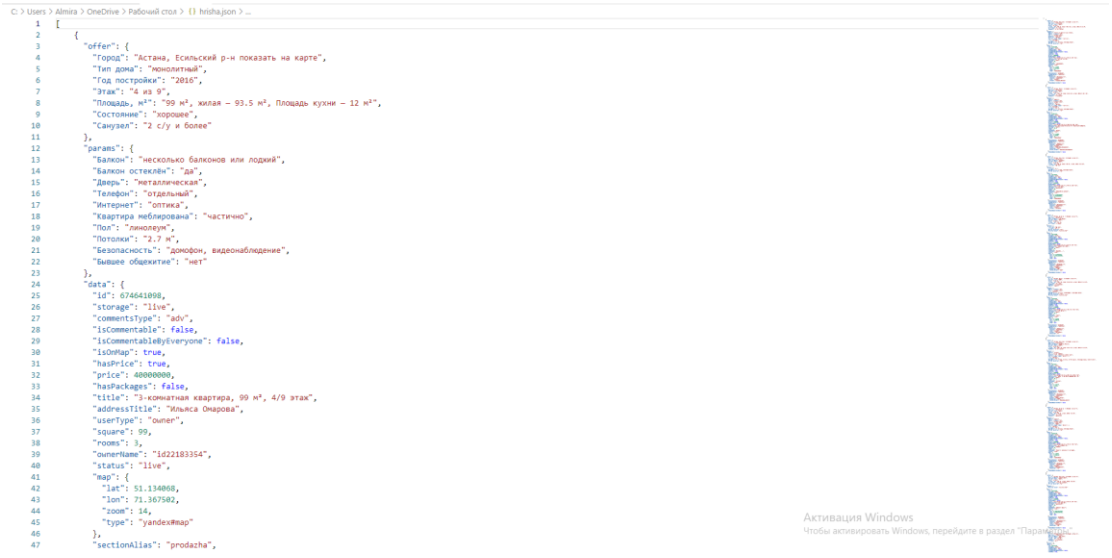
```
In [ ]: import json

with open('Data1.json') as user_file:
  file_contents = user_file.read()

print(file_contents)

parsed_json = json.loads(file_contents)
```

```
In [ ]: len(parsed_json)
```

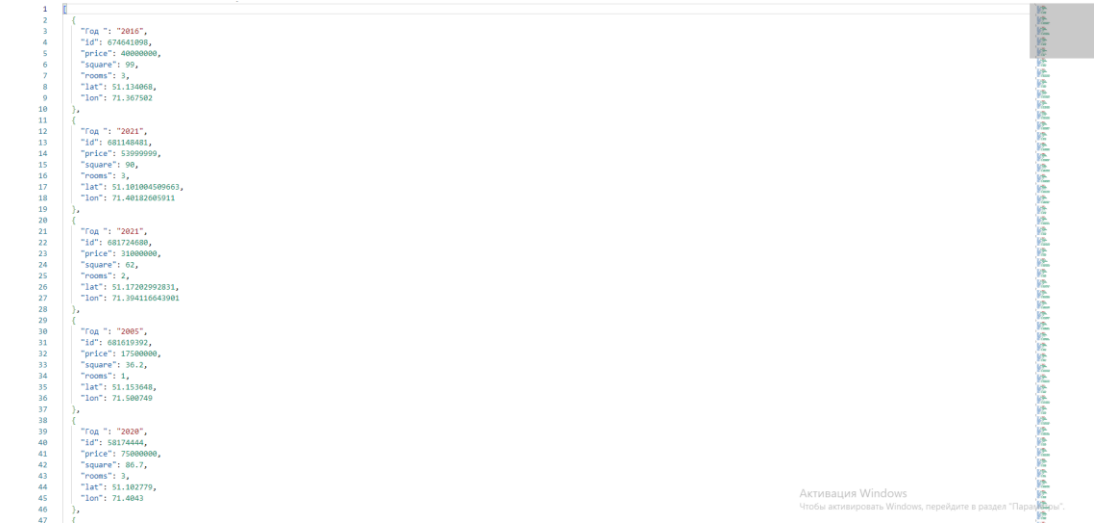
```
In [ ]: import json

with open('all.json') as user_file:
  file_contents = user_file.read()

print(file_contents)

parsed_json = json.loads(file_contents)
```

```
In [ ]: len(parsed_json)
```

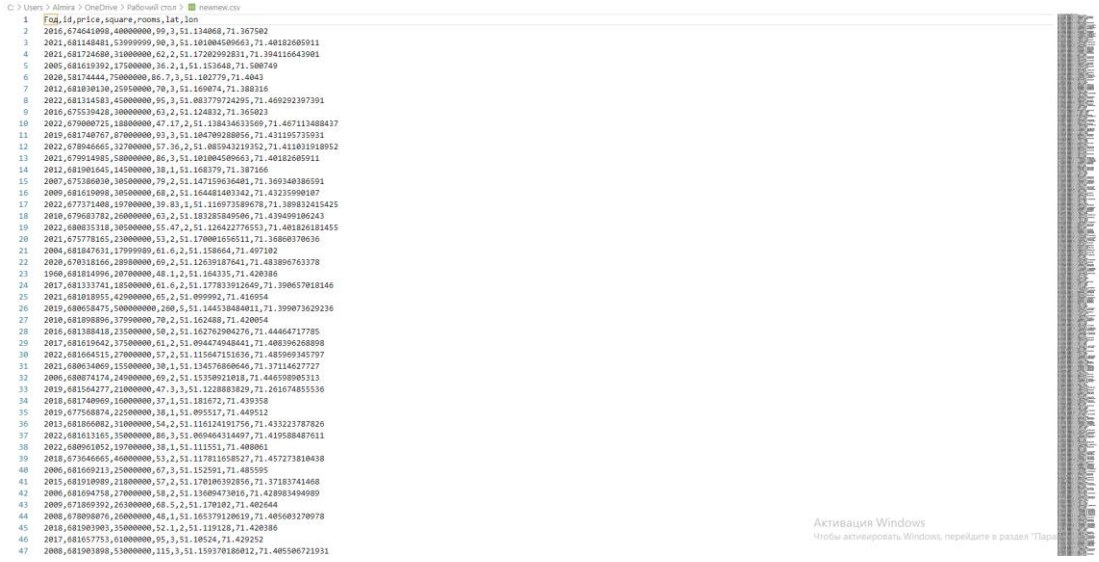
P.(1-3) Code for parsing.



P.4 How the data looked initially.



P.5 How the data looked after I corrected it.



P.6 How the data looked after formatting in CSV.

I used this site to convert a file from json to csv - https://www.convertcsv.com/json-to-csv.htm

	Год	id	price	square	rooms	lat	lon
count	2187.000000	2.187000e+03	2.187000e+03	2187.000000	2187.000000	2187.000000	2187.000000
mean	2013.202561	6.764468e+08	3.868365e+07	70.205688	2.121628	51.128470	71.431450
std	11.659222	4.685230e+07	4.599485e+07	46.516636	1.061177	0.239179	0.144116
min	1958.000000	1.363287e+07	5.700000e+06	13.000000	1.000000	43.238077	68.211424
25%	2010.000000	6.793429e+08	2.000000e+07	41.565000	1.000000	51.116350	71.395531
50%	2017.000000	6.811480e+08	2.700000e+07	58.100000	2.000000	51.133333	71.426216
75%	2021.000000	6.817389e+08	4.000000e+07	81.000000	3.000000	51.158096	71.465958
max	2022.000000	6.819111e+08	7.300000e+08	700.000000	11.000000	51.207335	76.898010

P.7 Analysis of the data.

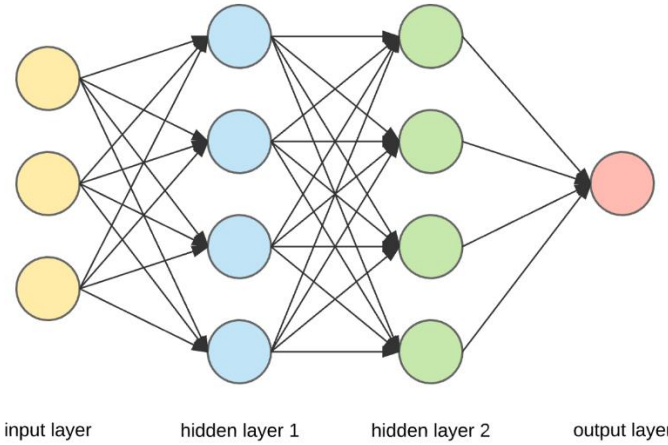
P.8 Data visualization (MongoDB): frequency by year.

P.9 Data visualization (MongoDB): frequency by number of rooms.

2.2 Description of the ML models you used with some theory.

I have created a sequential model for the project. It allows me to define a neural network sequentially, passing through several neural layers, one after the other. There are four layers in my model: 1 layer contains 4 inputs and the Relu activation function - which is one of the most common activation functions in machine learning models. Layers 2 and 3 also contain a Relu activation function. The final level will output the predicted value of the target variable. The ReLU activation functions help to introduce non-linearity into the model. The Model also

uses the standard error loss function (MSE), which measures the difference between the predicted and actual values of the target variable. The optimizer used is Adam, which is a popular optimization algorithm for neural networks. The goal of the optimizer is to adjust weights and offsets in the model during training to minimize the loss function. The Adam optimizer helps to ensure effective training.

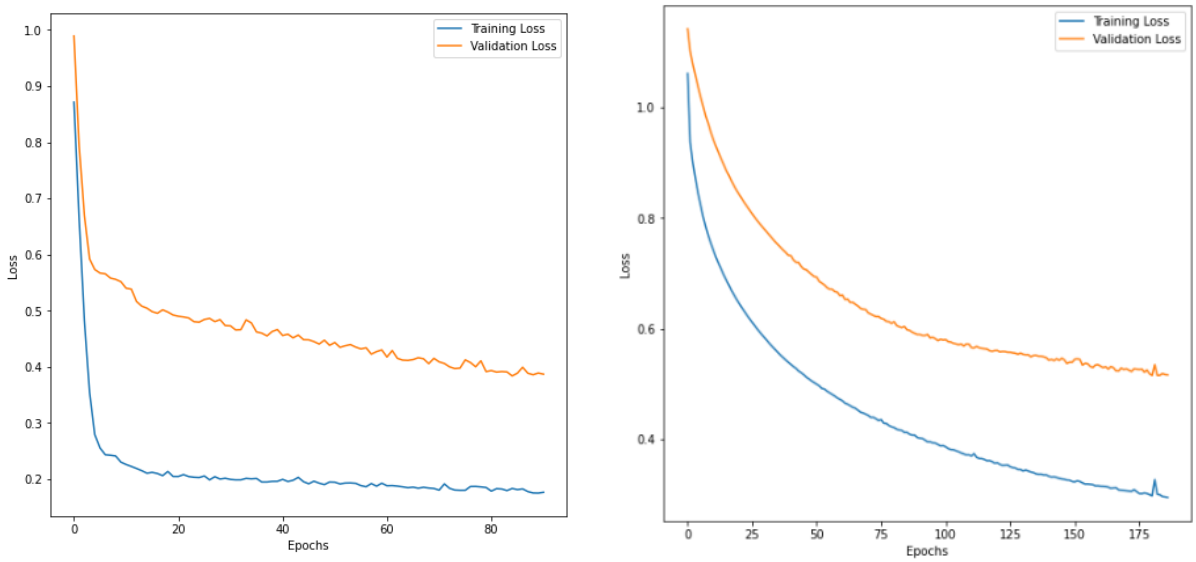


P.10 What a sequential model looks like in theory.

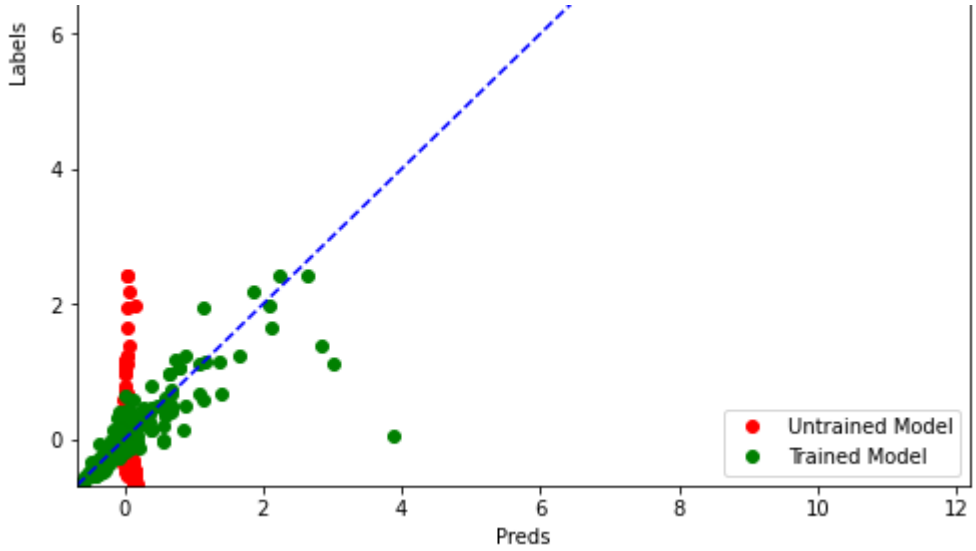
3. Result

3.1 Results with tables, pictures, and interesting numbers

I was able to find the accuracy of prediction from a personal dataset in my work, here you can see confirmation of this.



P. (11-12) My learning Curves (plotting the loss on the training set epoch by epoch.)



P. 13 Predicted Table

4. Discussion

4.1 Critical review of results

In my opinion, I did all the work that I could do using my modest dataset relative to other ready-made ones that are on the Internet, and also that I have low laptop power. Despite this, in the future, I want to increase the dataset, and use different cities for more detailed information.

4.2 Next steps

Using the acquired skills and knowledge, I can create and solve simple problems in the field of machine learning. Nevertheless in the future, I would like to improve my model and expand the data, including different cities in our country. I will continue to study this topic and try to move to a more difficult level.

5.

References

Alan Ihre & Isak Engstrom. (2019). *Predicting house prices with machine learning methods.* https://www.diva-portal.org/smash/get/diva2:1354741/FULLTEXT01.pdf.

Arden Dertat. (2017). *Applied Deep Learning, Part 1: Artificial Neural Networks.* https://towardsdatascience.com/applied-deep-learning-part-1-artificial-neural-networks-d7834f67a4f6.

ConvertCSV. (2023) *"JSON to CSV Converter."* https://www.convertcsv.com/json-to-csv.htm.

Kaggle. *Overfitting and Underfitting.* https://www.kaggle.com/code/ryanholbrook/overfitting-and-underfitting.

Keras. *EarlyStopping callback.* https://keras.io/api/callbacks/early_stopping/.

Sharmasapna. (2022). *House price prediction using The Ames Housing dataset.* https://github.com/sharmasapna/house-price-prediction

Shreyas Raghavan. (2017). *Create a model to predict house prices using Python*. https://github.com/Shreyas3108/house-price-prediction.

Stats Wire. (2022). *House Price Prediction Regression | Python | TensorFlow*. https://www.youtube.com/watch?v=N942Bi0_FnI.

TahaSherif. (2020). *Predicting House Prices with Regression using Tensorflow.* https://github.com/TahaSherif/Predicting-House-Prices-with-Regression-Tensorflow.

Techopedia. (2020). *Rectified Linear Unit (ReLU).* https://www.techopedia.com/definition/33346/rectified-linear-unit-relu#:~:text=The%20rectified%20linear%20unit%20(ReLU,helping%20to%20deliver%20an%20output.

Tensordroid. (2021). *House Price Prediction Model overview using Tensorflow*. https://www.youtube.com/watch?v=90xKZBZWbKg.