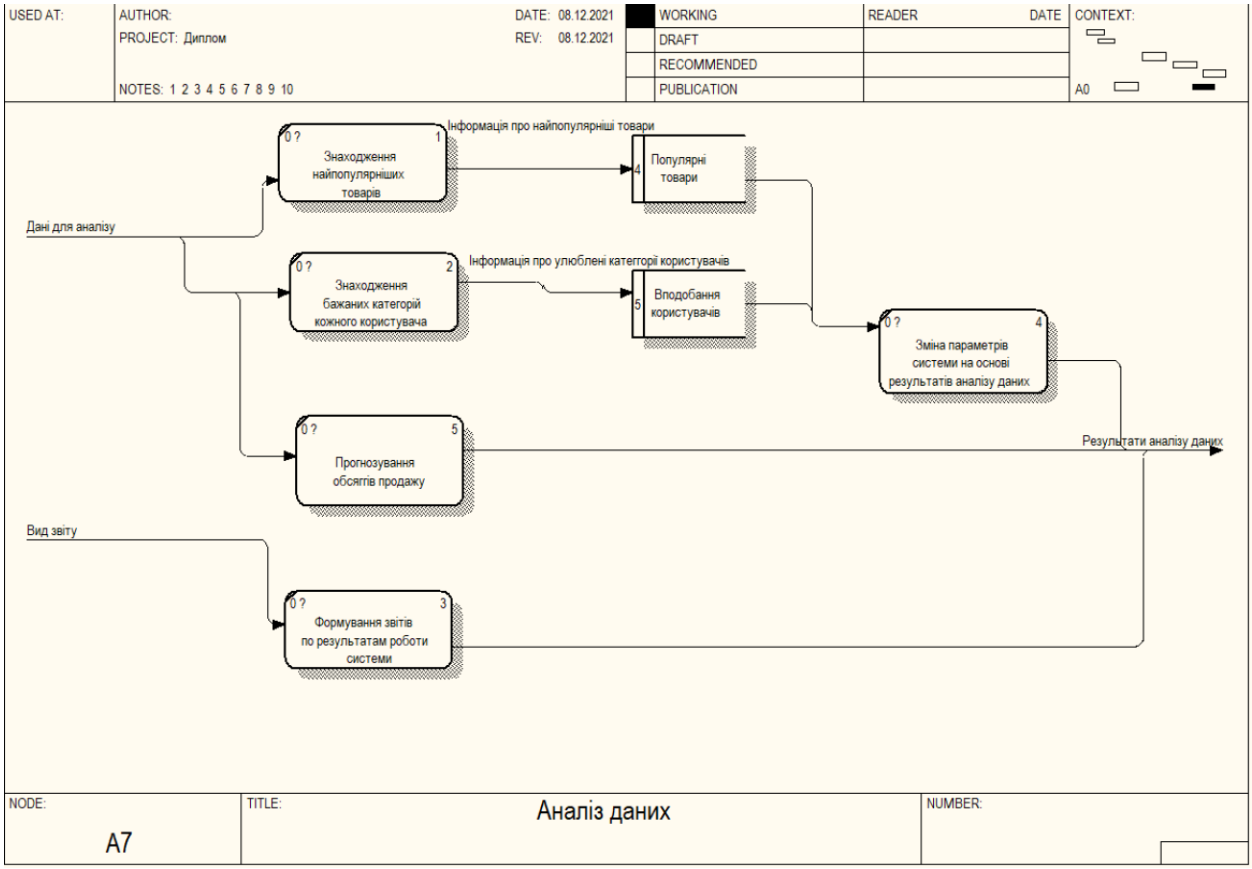
Додаток А

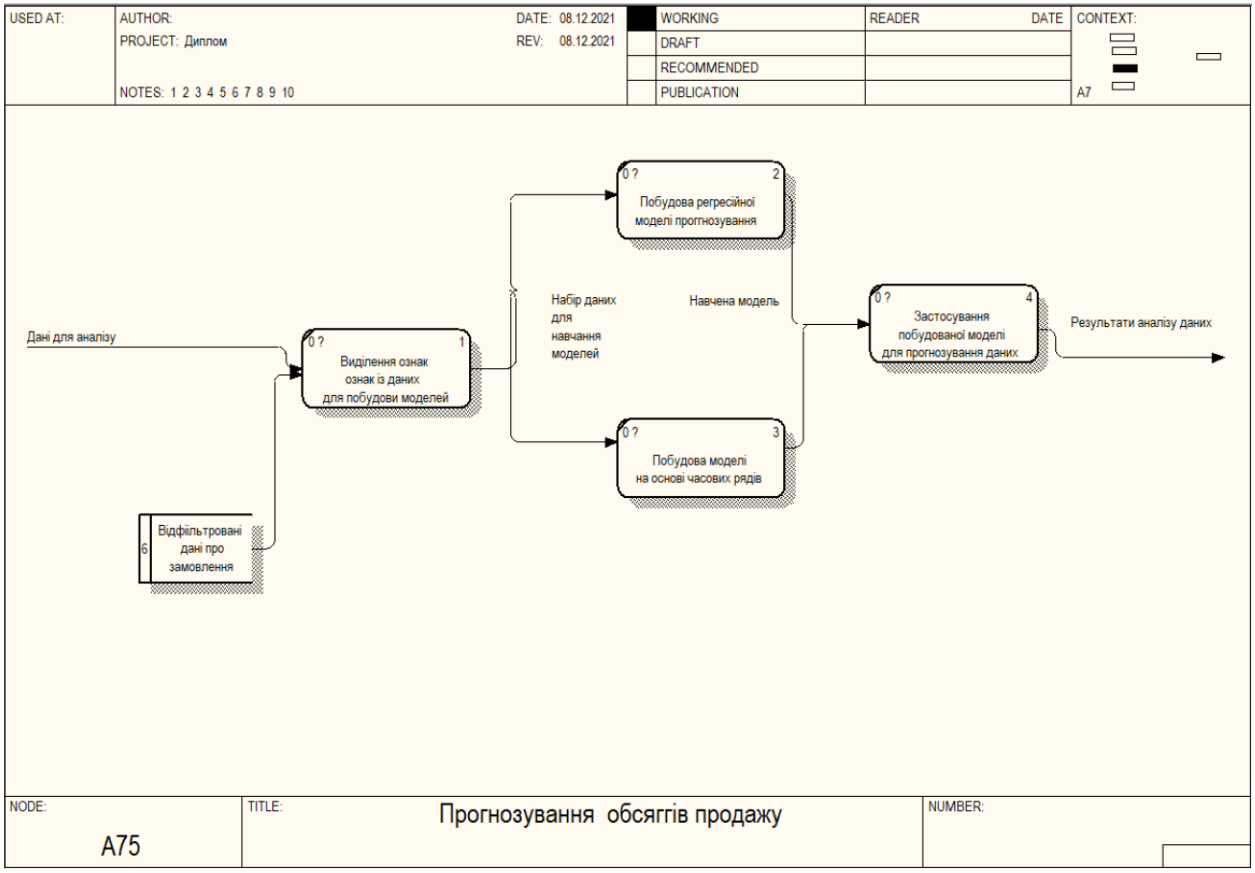
Графічні матеріали

ДЕКОМПОЗИЦІЯ ПРОЦЕСУ «АНАЛІЗ ДАНИХ»



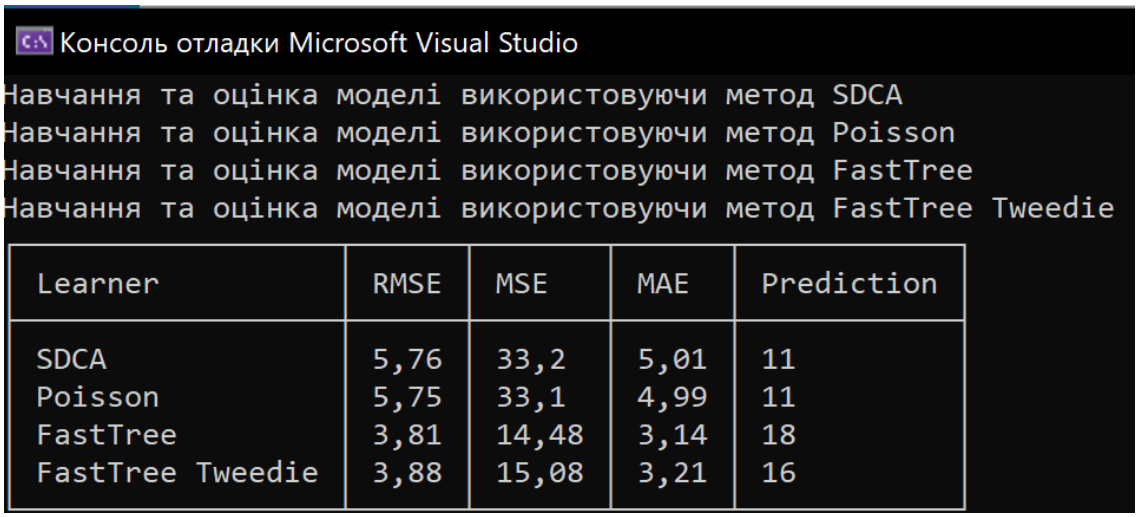
Розроб.	*Гайдар М.І.*		*15.12.21*	*Дослідження застосування методів аналізу даних в системах електронної комерції*	
Перевір.	*Калита Н.І.*		*15.12.21*	*СПРм-20-1*	*Аркуш 1*
Н. Контр.	*Калита Н.І.*		*15.12.21*	*СТ*	*Аркушів 1*
Затверд.	*Гребеннік І.В.*				

ДЕКОМПОЗИЦІЯ ПРОЦЕСУ «ПРОГНОЗУВАННЯ ДАНИХ»



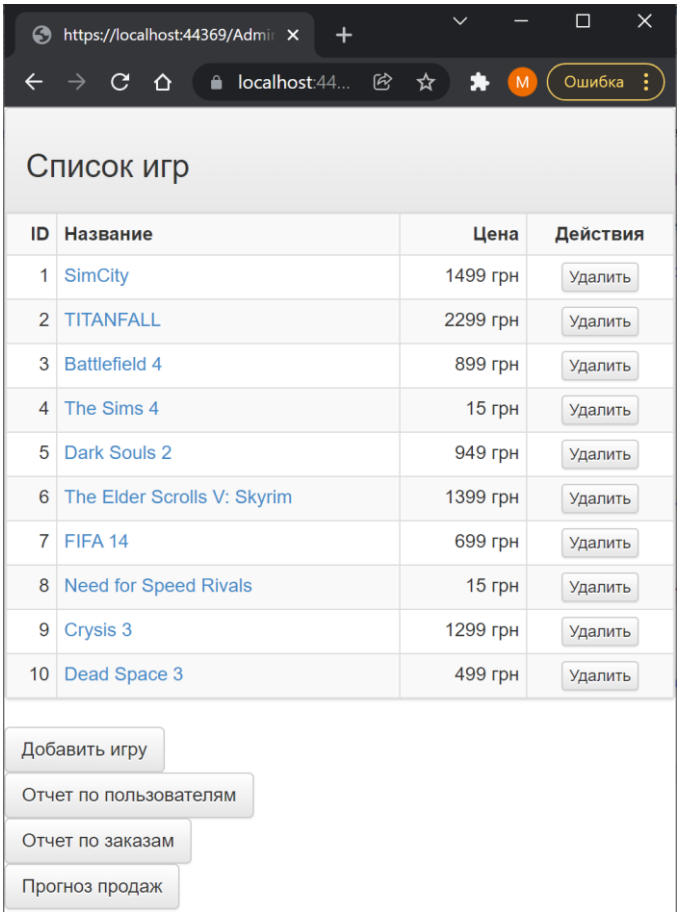
Розроб.	*Гайдар М.І.*		*15.12.21*	*Дослідження застосування методів аналізу даних в системах електронної комерції*	
Перевір.	*Калита Н.І.*		*15.12.21*	*СПРм-20-1*	*Аркуш 1*
Н. Контр.	*Калита Н.І.*		*15.12.21*	*СТ*	*Аркушів 1*
Затверд.	*Гребеннік І.В.*				

ОЦІНКА ПОБУДОВАНИХ МОДЕЛЕЙ РЕГРЕСІЇ



Learner	RMSE	MSE	MAE	Prediction
SDCA	5,76	33,2	5,01	11
Poisson	5,75	33,1	4,99	11
FastTree	3,81	14,48	3,14	18
FastTree Tweedie	3,88	15,08	3,21	16

СТОРІНКА ПАНЕЛІ АДМІНІСТРАТОРА



Розроб.	*Гайдар М.І.*		*15.12.21*	*Дослідження застосування методів аналізу даних в системах електронної комерції*	
Перевір.	*Калита Н.І.*		*15.12.21*	*СПРм-20-1*	*Аркуш 1*
Н. Контр.	*Калита Н.І.*		*15.12.21*	*СТ*	*Аркушів 1*
Затверд.	*Гребеннік І.В.*				

РЕЗУЛЬТАТИ ПРОГНОЗУВАННЯ ОБСЯГІВ ПРОДАЖІВ ДЛЯ КАТЕГОРІЇ «RPG» І ЦІНОВОГО ДІАПАЗОНУ «ДО 500 ГРН.»

Прогнозирование продаж

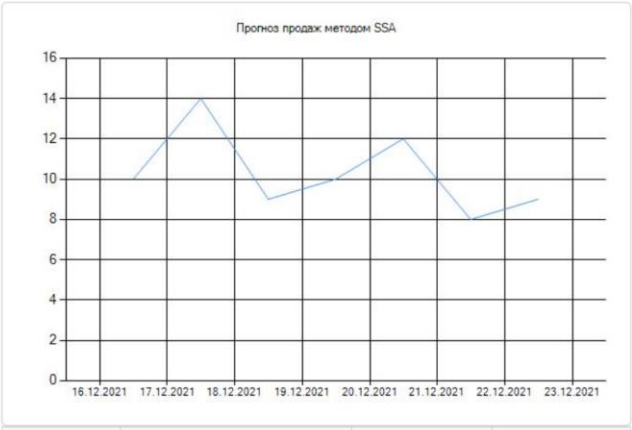
RPG | До 500 грн. | Получить

Прогноз продаж на

категорию RPG

и

ценовой диапазон До 500 грн.



Дата	Ориентировочное кол-во продаж	Категория товаров	Ценовая категория
16.12.2021	10 штук	RPG	До 500 грн.
17.12.2021	14 штук	RPG	До 500 грн.
18.12.2021	9 штук	RPG	До 500 грн.
19.12.2021	10 штук	RPG	До 500 грн.
20.12.2021	12 штук	RPG	До 500 грн.
21.12.2021	8 штук	RPG	До 500 грн.
22.12.2021	9 штук	RPG	До 500 грн.

Дата	Ориентировочное кол-во продаж	Категория товаров	Ценовая категория
16.12.2021	12 штук	RPG	До 500 грн.
17.12.2021	13 штук	RPG	До 500 грн.
18.12.2021	7 штук	RPG	До 500 грн.
19.12.2021	7 штук	RPG	До 500 грн.
20.12.2021	8 штук	RPG	До 500 грн.
21.12.2021	9 штук	RPG	До 500 грн.
22.12.2021	10 штук	RPG	До 500 грн.

Додаток Б

Текст програми

01 12 01

Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

«ЗАТВЕРДЖУЮ»

Керівник кваліфікаційної роботи

_____ проф. Калита Н.І..
(підпис)

ДОСЛІДЖЕННЯ ЗАСТОСУВАННЯ МЕТОДІВ АНАЛІЗУ ДАНИХ В СИСТЕМАХ ЕЛЕКТРОННОЇ КОМЕРЦІЇ

Текст програми

ЛИСТ ЗАТВЕРДЖЕННЯ

01 12 01 -ЛЗ

РОЗРОБИВ

Ст. гр. СПРм-20-1

Гайдар М.І.

2021

ЗАТВЕРДЖЕНО

01 12 01 -ЛЗ

ДОСЛІДЖЕННЯ ЗАСТОСУВАННЯ МЕТОДІВ АНАЛІЗУ ДАНИХ В СИСТЕМАХ ЕЛЕКТРОННОЇ КОМЕРЦІЇ

Текст програми

01 12 01

Аркушів 14

2021

Загальні відомості

Даний код реалізує компонент IC електронної комерції, який відповідає за прогнозування обсягів продажів використовуючи методи бібліотеки машинного навчання Microsoft.ML. При реалізації функції були використані C #, HTML, СУБД MySQL.

This code implements the e-commerce IP component, which is responsible for forecasting sales using Microsoft.ML machine learning library methods. C #, HTML, MySQL DBMS were used in the implementation of the function.

Код програми:

Представлення макету сторінки прогнозування:

```
@model Gs.WebUI.Models.PredModel
@{
    ViewBag.Title = "Прогноз продаж";
    Layout = "~/Views/Shared/_AdminLayout.cshtml";
}

<!DOCTYPE html>

<div class="panel panel-default">
    <form>
        <div class="row" style="margin-right: 10px" align="center">
            <h3>Прогнозирование продаж</h3>
            <p>
                <select name="category">
                    <option>Выберите категорию</option>
                    <option value="1">RPG</option>
                    <option value="2">Симуляторы</option>
                    <option value="3">Шутеры</option>
                </select>
                <select name="price">
                    <option>Выберите ценовой диапазон</option>
                    <option value="1">До 500 грн.</option>
                    <option value="2">От 500 до 1500 грн</option>
                    <option value="3">Больше 1500 грн</option>
                </select>
                <input type="submit" value="Получить" formmethod="post">

        </div>
    </form>
</div>

@if (Model != null)
{
    <div class="row" style="margin-right: 10px" align="center">

        @if (Model.c == 0 && Model.p == 0)
        { }
        else
        {
```

```
            <h3>Прогноз продаж на </h3>
            {
                switch (Model.c)
                {
                    case 0:
                        break;
                    case 1:
                        <h3>категорию RPG</h3>
                        break;
                    case 2:
                        <h3>категорию Симулятор</h3>
                        break;
                    case 3:
                        <h3>категорию Шутер</h3>
                        break;
                }
            }
            if (Model.p != 0)
            {
                <h3>и</h3>}
            {
                switch (Model.p)
                {
                    case 0:
                        break;
                    case 1:
                        <h3> ценовой диапазон До 500 грн.</h3>
                        break;
                    case 2:
                        <h3> ценовой диапазон От 500 до 1500 грн</h3>
                        break;
                    case 3:
                        <h3> ценовой диапазон Больше 1500 грн</h3>
                        break;
                }
            }
        }
    </div>


        <div class="col-xs-6">

            <img class="img-thumbnail" width="600" height="400"
                 src="~/Admin/predictSSA.jpeg" />

            <table class="table table-striped  table-bordered" align="center"
width="50%">
                <tr>
                    <th class="text-center">Дата</th>
                    <th class="text-center">Ориентировочное кол-во продаж</th>
                    <th class="text-center">Категория товаров</th>
                    <th class="text-center">Ценовая категория</th>
                </tr>

                @foreach (var item in Model.PredictionSSA)
                {
                    <tr>
                        <td class="text-center">@item.RentalDate.ToShortDateString()</td>
                        <td class="text-center">@item.TotalRentals.ToString("#
штук")</td>
                        @{switch (item.categ)
                            {
                                case 1:
                                    <td class="text-center">RPG</td>
                                    break;
```

```
                                case 2:
                                    <td class="text-center">Симулятор</td>
                                    break;
                                case 3:
                                    <td class="text-center">Шутер</td>
                                    break;
                            }
                        }
                        @{switch (item.p)
                            {
                                case 1:
                                    <td class="text-center">До 500 грн.</td>
                                    break;
                                case 2:
                                    <td class="text-center">От 500 до 1500 грн</td>
                                    break;
                                case 3:
                                    <td class="text-center">Больше 1500 грн</td>
                                    break;
                            }
                        }
                    </tr>
                }
            </table>
        </div>


        <div class="col-xs-6">

            <img class="img-thumbnail" width="600" height="400"
                 src="~/Admin/predictFTT.jpeg" />

            <table class="table table-striped  table-bordered" align="center"
width="50%">
                <tr>
                    <th class="text-center">Дата</th>
                    <th class="text-center">Ориентировочное кол-во продаж</th>
                    <th class="text-center">Категория товаров</th>
                    <th class="text-center">Ценовая категория</th>
                </tr>

                @foreach (var item in Model.PredictionFTT)
                {
                    <tr>
                        <td class="text-center">@item.RentalDate.ToShortDateString()</td>
                        <td class="text-center">@item.TotalRentals.ToString("#
штук")</td>
                        @{switch (item.categ)
                            {
                                case 1:
                                    <td class="text-center">RPG</td>
                                    break;
                                case 2:
                                    <td class="text-center">Симулятор</td>
                                    break;
                                case 3:
                                    <td class="text-center">Шутер</td>
                                    break;
                            }
                        }
                        @{switch (item.p)
                            {
                                case 1:
                                    <td class="text-center">До 500 грн.</td>
                                    break;
```

```
                                case 2:
                                    <td class="text-center">От 500 до 1500 грн</td>
                                    break;
                                case 3:
                                    <td class="text-center">Больше 1500 грн</td>
                                    break;
                            }
                        }
                    </tr>
                }
            </table>
        </div>

}
```

Метод класу AdminController який відповідає за прогнозування:

```
public ActionResult Pred()
        {
            return View();
        }



        [HttpPost]
        public ActionResult Pred(int category=0, int price=0)
        {

            string rootDir =
Path.GetFullPath(Path.Combine(AppDomain.CurrentDomain.BaseDirectory, "../../../"));
            string modelPath = Path.Combine(rootDir, "MLModel.zip");
            string connectionString =
"server=localhost;user=root;database=gamestore;port=3306;password=root";

            MySqlConnection conn = new MySqlConnection(connectionString);

            MLContext mlContext = new MLContext();
```

```
            DatabaseLoader loader = mlContext.Data.CreateDatabaseLoader<ModelInput>();

            DatabaseLoader loaderFTT =
mlContext.Data.CreateDatabaseLoader<ModelInputFF>();


            string query = "SELECT RentalDate, Year, TotalRentals, q, categ, p FROM
info1";


            DatabaseSource dbSource = new DatabaseSource(MySqlClientFactory.Instance,

                                connectionString,

                                query);


            IDataView dataView = loader.Load(dbSource);


            IDataView firstYearData = mlContext.Data.FilterRowsByColumn(dataView, "Year",
upperBound: 1);

            IDataView secondYearData = mlContext.Data.FilterRowsByColumn(dataView,
"Year", lowerBound: 1);




            query = "SELECT year(RentalDate) as yearr, month(RentalDate) as month,
dayofmonth(RentalDate) as day, Year, TotalRentals as Label, q, categ, p FROM info1";


            DatabaseSource dbSource1 = new DatabaseSource(MySqlClientFactory.Instance,

                                connectionString,

                                query);


            IDataView trainingData = loaderFTT.Load(dbSource1);


            var pipeline = mlContext.Transforms.Concatenate(

               "Features",

               nameof(ModelInputFF.yearr),

               nameof(ModelInputFF.month),

               nameof(ModelInputFF.day),

               nameof(ModelInputFF.Year),
```

```
                nameof(ModelInputFF.q),
                nameof(ModelInputFF.categ),
                nameof(ModelInputFF.p))
                // step 2: cache the data to speed up training
                .AppendCacheCheckpoint(mlContext);

            var fullPipeline =
pipeline.Append(mlContext.Regression.Trainers.FastTreeTweedie());

            var trainedModel = fullPipeline.Fit(trainingData);

            var forecastingPipeline = mlContext.Forecasting.ForecastBySsa(
            outputColumnName: "ForecastedRentals",
            inputColumnName: "TotalRentals",
            windowSize: 63,
            seriesLength: 270,
            trainSize: 365,
            horizon: 63,
            confidenceLevel: 0.95f,
            confidenceLowerBoundColumn: "LowerBoundRentals",
            confidenceUpperBoundColumn: "UpperBoundRentals");

            SsaForecastingTransformer forecaster =
forecastingPipeline.Fit(firstYearData);

            DateTime buf = DateTime.Now;
            List<ModelInput> ListSSA = new List<ModelInput>();
            DateTime mark = buf.AddDays(7);
            int days = 7;
            do
            {
                buf = buf.AddDays(1);
```

```
    for (int i = 1; i < 4; i++)
    {

        ListSSA.Add(new ModelInput
        {
            RentalDate = buf,
            Year = 3,
            TotalRentals = 0,
            q = (buf.Month + 2) / 3,
            categ = i,
            p = 2
        });

    }

}
while (buf.Date.CompareTo(mark.Date) != 0);

List<ModelInput> predDataSSA = new List<ModelInput>();

foreach (ModelInput row in ListSSA.ToList())
{
    for (int i = 1; i < 4; i++)
    {
        predDataSSA.Add(new ModelInput
        {
            RentalDate = row.RentalDate,
            Year = row.Year,
            TotalRentals = 0,
            q = (row.RentalDate.Month + 2) / 3,
            categ = row.categ,
            p = i
```

```
        });
    }
}

buf = DateTime.Now;
List<ModelInputFF> ListFF = new List<ModelInputFF>();
do
{
    buf = buf.AddDays(1);

    for (int i = 1; i < 4; i++)
    {

        ListFF.Add(new ModelInputFF
        {
            yearr = buf.Year,
            month = buf.Month,
            day = buf.Day,
            Year = 3,
            TotalRentals = 0,
            q = (buf.Month + 2) / 3,
            categ = i,
            p = 2
        });

    }

}
while (buf.Date.CompareTo(mark.Date) != 0);

List<ModelInputFF> predDataFTT = new List<ModelInputFF>();
```

```
            foreach (ModelInputFF row in ListFF.ToList())
            {
                for (int i = 1; i < 4; i++)
                {
                    predDataFTT.Add(new ModelInputFF
                    {
                        yearr = row.yearr,
                        month = row.month,
                        day = row.day,
                        Year = row.Year,
                        TotalRentals = 0,
                        q = ((int)row.month + 2) / 3,
                        categ = row.categ,
                        p = i
                    });
                }
            }

            var engine = mlContext.Model.CreatePredictionEngine<ModelInputFF,
DemandPrediction>(trainedModel);

            foreach (ModelInputFF sample in predDataFTT)
            {
                var prediction = engine.Predict(sample);
                sample.TotalRentals = prediction.PredictedCount;
            }

            List<ModelInput> predDataFFTN = new List<ModelInput>();


            foreach (ModelInputFF row in predDataFTT)
            {
                predDataFFTN.Add(new ModelInput
```

```
                {
                    RentalDate = new DateTime((int)row.yearr, (int)row.month,
(int)row.day),
                    Year = row.Year,
                    TotalRentals = row.TotalRentals,
                    q = row.q,
                    categ = row.categ,
                    p = row.p
                });
            }

            IDataView test =
mlContext.Data.LoadFromEnumerable(predDataSSA.AsEnumerable());

            var forecastEngine = forecaster.CreateTimeSeriesEngine<ModelInput,
ModelOutput>(mlContext);

            forecastEngine.CheckPoint(mlContext, modelPath);

            ModelOutput forecast = forecastEngine.Predict();

            IEnumerable<double> forecastOutput =
            mlContext.Data.CreateEnumerable<ModelInput>(test, reuseRowObject: false)
                .Take(days*9 + 1)
                .Select((ModelInput rental, int index) =>
                {
                    double estimate = forecast.ForecastedRentals[index];
                    return Math.Round(estimate);
                });

            List<double> res = forecastOutput.ToList();

            for (int i = 0; i < predDataSSA.Count(); i++)
```

```
            {
                predDataSSA[i].TotalRentals = (float)res[i];
            }

            IEnumerable<ModelInput> predDataSSAFilt;
            IEnumerable<ModelInput> predDataFFTFilt;

            if (category != 0 && price != 0)
            {
                predDataSSAFilt = predDataSSA.Where(m => m.categ == category && m.p ==
price)
                  .GroupBy(x => x.RentalDate, (key, group) => new ModelInput
                  {
                      RentalDate = key,
                      Year = group.First().Year,
                      q = group.First().q,
                      categ = group.First().categ,
                      p = group.First().p,
                      TotalRentals = group.Sum(m => m.TotalRentals)
                  });

                predDataFFTFilt = predDataFFTN.Where(m => m.categ == category && m.p ==
price)
                  .GroupBy(x => x.RentalDate, (key, group) => new ModelInput
                  {
                      RentalDate = key,
                      Year = group.First().Year,
                      q = group.First().q,
                      categ = group.First().categ,
                      p = group.First().p,
                      TotalRentals = group.Sum(m => m.TotalRentals)
                  });
            }
```

```
else if (category != 0 && price == 0)
{
    predDataSSAFilt = predDataSSA.Where(m => m.categ == category)
        .GroupBy(x => x.RentalDate, (key, group) => new ModelInput
        {
            RentalDate = key,
            Year = group.First().Year,
            q = group.First().q,
            categ = group.First().categ,
            p = group.First().p,
            TotalRentals = group.Sum(m => m.TotalRentals)
        });

    predDataFFTFilt = predDataFFTN.Where(m => m.categ == category)
      .GroupBy(x => x.RentalDate, (key, group) => new ModelInput
      {
          RentalDate = key,
          Year = group.First().Year,
          q = group.First().q,
          categ = group.First().categ,
          p = group.First().p,
          TotalRentals = group.Sum(m => m.TotalRentals)
      });
}
else if (category == 0 && price != 0)
{
    predDataSSAFilt = predDataSSA.Where(m => m.p == price)
        .GroupBy(x => x.RentalDate, (key, group) => new ModelInput
        {
            RentalDate = key,
            Year = group.First().Year,
            q = group.First().q,
```

```
            categ = group.First().categ,
            p = group.First().p,
            TotalRentals = group.Sum(m => m.TotalRentals)
        });

    predDataFFTFilt = predDataFFTN.Where(m => m.p == price)
      .GroupBy(x => x.RentalDate, (key, group) => new ModelInput
      {
          RentalDate = key,
          Year = group.First().Year,
          q = group.First().q,
          categ = group.First().categ,
          p = group.First().p,
          TotalRentals = group.Sum(m => m.TotalRentals)
      });
}
else
{
    predDataSSAFilt = predDataSSA;
    predDataFFTFilt = predDataFFTN;
}

List<ModelInput> ChartData =predDataSSAFilt.ToList();


DateTime[] preddate = new DateTime[ChartData.Count()];
int[] sales = new int[ChartData.Count()];

for (int i = 0; i < predDataSSAFilt.Count(); i++)
{
    preddate[i] = ChartData[i].RentalDate;
    sales[i] = (int)ChartData[i].TotalRentals;
```

```
            }
            var myChart1 = new Chart(width: 600, height: 400)
                .AddTitle("Прогноз продаж методом SSA")
                .AddSeries(chartType: "Line",
                    name: "sales",
                    xValue: preddate,
                    yValues: sales).Save(".\\predictSSA.jpeg", "jpeg");

            ChartData = predDataFFTFilt.ToList();



            for (int i = 0; i < ChartData.Count(); i++)
            {
                preddate[i] = ChartData[i].RentalDate;
                sales[i] = (int)ChartData[i].TotalRentals;
            }

            var myChart2 = new Chart(width: 600, height: 400)
                .AddTitle("Прогноз продаж методом FastTree")
                .AddSeries(chartType: "Line",
                    name: "sales",
                    xValue: preddate,
                    yValues: sales).Save(".\\predictFTT.jpeg", "jpeg");



            PredModel results = new PredModel { PredictionSSA = predDataSSAFilt,
PredictionFTT = predDataFFTFilt, c=category, p=price };



            return View(results);
```

Відомість кваліфікаційної роботи

№	Позначення	Найменування	Примітки
1.		Пояснювальна записка	75 арк.
2.		Текст програми	14 арк.
		Графічні документи	
3.		Декомпозиція функції	1 аркуш
		«Аналіз даних»	
4.		Декомпозиція функції	1 аркуш
		«Прогнозування даних»	
5.		Оцінка побудованих	1 аркуш
		моделей регресії	
6.		Сторінка панелі	1 аркуш
		адміністратора	
7.		Результати прогнозування	1 аркуш
		обсягів продажів для категорії	
8.		«RPG» і цінового діапазону	
		«до 500 грн.»	

					ДЗ			
Ізм.	*Арк*	*№ докум.*	*Підпис*	*Дата*				
Рорзобив		*Гайдар М.І.*		*15.12.21*	*Дослідження застосування методів аналізу даних в системах електронної комерції*	*Літера*	Арк.	Аркушів
Перевірив.		*Калита Н.І.*		*15.12.21*			*1*	*1*
Н. Контр.		*Калита Н.І.*		*15.12.21*		*ХНУРЕ Кафедра СТ*		
Затвер.		*Гребенник И.В.*						